COMMUNITY DETECTION IN CENSORED HYPERGRAPH

A Dissertation
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Bin Zhao

In Partial Fulfillment of the Requirements
for the Degree of
DOCTOR OF PHILOSOPHY

Major Department:
Statistics

October 2023

Fargo, North Dakota

NORTH DAKOTA STATE UNIVERSITY

Graduate School

Title

COMMUNITY DETECTION IN CENSORED HYPERGRAPH

By

Bin Zhao

The supervisory committee certifies that this dissertation complies with North Dakota State University's regulations and meets the accepted standards for the degree of

DOCTOR OF PHILOSOPHY

SUPERVISORY COMMITTEE:

Prof. Mingao Yuan
Chair

Prof. Megan Orr

Prof. Bong-Jin Choi

Prof. Lu Liu

Prof. Zhaohui Liu

Approved:

14 November 2023
Date

Prof. Rhonda Magel
Department Chair

ABSTRACT

Network, or graph, represent relationships between entities in various applications, such as social networks, biological systems, and communication networks. A common feature in network data is the presence of community structures, where groups of nodes exhibit higher connectivity within themselves than with other groups. Identifying these community structures, a task known as community detection, is essential for gaining valuable insights in diverse applications, including uncovering hidden relationships in social networks, detecting functional modules in biological systems, and identifying vulnerabilities in communication networks. However, real-world network data may have missing values, significantly impacting the network's structural properties. Existing community detection methods primarily focus on networks without missing values, leaving a gap in the analysis of censored networks. This study addresses the community detection problem in censored m-uniform hypergraphs. Firstly, utilizing an information-theoretic approach, we obtain a threshold that enables the exact recovery of the community structure. Then, we proposed a two-stage polynomial-time algorithm, which encompasses a spectral algorithm complemented by a refinement step, aiming to achieve exact recovery. Moreover, we introduce a semi-definite relaxation algorithm, studying its operational performance as a standalone community detection algorithm, without the integration of a refinement step. Lastly, in consideration of the effect of imputation methods on censored hypergraphs, we propose several methods grounded in network properties. We subsequently employ simulation to assess the performance of these methods. Finally, we apply the proposed algorithm to real-world data, showcasing its practical utility in various settings.

ACKNOWLEDGEMENTS

Completing this Ph.D. dissertation has been a profound journey, one that would not have been possible without the guidance, support, and encouragement of many. I am immensely grateful to a number of individuals whose contributions were invaluable to my academic pursuit.

Foremost, I extend my deepest gratitude to my advisor, Dr. Mingao Yuan, whose expertise and insightful critiques were instrumental in sculpting this research. His unwavering support and patience throughout the research process have been pillars of strength for me.

I am also indebted to my committee members, Dr. Megan Orr, Dr. Bong-Jin Choi, Dr. Lu Liu, and Dr. Zhaohui Liu, as well as Dr. Lauren Hanna, for their invaluable feedback and rigorous standards, which significantly contributed to the depth and quality of this work and my previous research. Their willingness to share knowledge and invest time has enriched my learning experience profoundly.

To my family, who has provided me with an unwavering foundation of love and support, I owe an immense debt of gratitude. A special mention goes to my wife, Xin Xin, whose love, patience, and encouragement have been my sanctuary. Her belief in me often gave me the strength to persevere when challenges arose.

My time in Fargo has been made all the more memorable by friends who have become like family. I am particularly thankful to Yun Zhou, Tong Lin, and all the people from Red River Valley Chinese Christian Church, whose kindness and readiness to help at any moment have been a source of comfort and joy. The selflessness and camaraderie they have shown have made all the difference during my studies.

To all my friends, colleagues, and the academic community at Fargo, who have contributed to my journey in ways big and small, I am forever appreciative.

Lastly, I acknowledge all those who have indirectly influenced this work with their presence and goodwill. To everyone who has been a part of my journey, thank you for being the trellis on which I have grown academically and personally.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

1.1. Literature Review

A simple graph, denoted as $G = (V, E)$, is a mathematical structure comprised of two finite sets: the vertex set V and the edge set E. The vertex set V is non-empty and contains distinct elements known as vertices or nodes, while the edge set E consists of distinct unordered pairs of distinct vertices. An exemplification of such a structure is depicted in Figure 1.1. In this graph, the vertex set is $V = \{1, 2, 3, 4, 5, 6\}$, and the edge set is $E = \{(1, 2), (1, 5), (2, 3), (2, 5), (3, 4), (4, 5), (4, 6)\}$.

Networks have found extensive applications across diverse fields (Chen and Yuan (2006); Costa et al. (2011); Fortunato, S. (2010)). In application, vertices symbolize independent entities and edges represent the relationships or interactions between these entities. For instance, in transportation systems, a network graph might comprise locations (nodes) and routes (edges) (De Bona et al. (2021)). In biological systems, we can construct a protein-protein interaction network wherein proteins are represented as nodes and pairs of physical interactions between these proteins as edges. Similarly, we can model the internet as a graph, with webpages serving as nodes and hyperlinks between pages forming edges (Ángeles et al. (2006)). Due to wide applications, network analysis has become a focal point in statistical and machine learning communities, spawning a rich array of research.

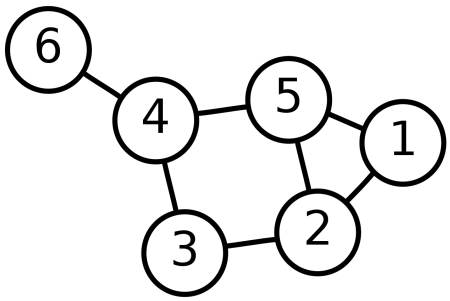

Figure 1.1. An undirected graph

One of the most popular topics in network data mining is to understand which items are similar to each other. This concept is grounded on the premise of a community structure (Abbe

(2018)), where nodes within the same community are densely connected, while connections between different communities are sparse. In essence, community structure reflects a high degree of homogeneity within communities and heterogeneity between them. Concretely, Figure 1.2 illustrates the community structure within a graph comprised of 100 vertices. In the graph on the left, no community structure exists, as the edges are uniformly distributed without any discernible clustering. Conversely, the graph on the right displays two distinct communities, with nodes of the same color forming a single community. It's evident that nodes within the same community exhibit denser connections in comparison to nodes across different communities, highlighting the presence of a well-defined community structure.

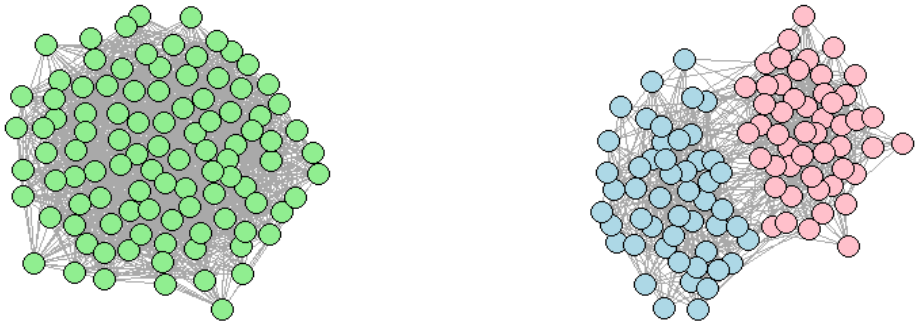

Figure 1.2. Graphs sampled from stochastic block model. Note: we sampled the nodes in the left graph from a uniform distribution, and sampled the nodes in the right graph with a within-cluster probability of 3/10 and an across-cluster probability of 1/20. The aforementioned probabilities are the probabilities that the nodes from the same community and nodes from different communities can form an edge.

Community detection (Abbe (2018); Amini et al. (2013); Bickel, P. J., and Sarkar, P. (2016)), seeks to discern the inherent structure in networks by grouping similar nodes into communities. Community detection is widely used in analyses of social networks (Goldenberg et al. (2010); Zhao et al. (2011)). Community detection in social networks identifies groups of nodes (representing individuals or organizations) with strong connections within the group and weak connections to the rest of the network. This process facilitates the revelation of underlying structural patterns and relationships within social networks. It enables the pinpointing of influential nodes and improves the performance of recommendation systems. Community detection is an invaluable tool in the

realm of biological sciences, particularly in the analysis of protein-to-protein interaction networks (Chen and Yuan (2006)). In such networks, proteins serve as nodes, while the edges denote physical or functional interactions between them, allowing for a deeper understanding of protein function and cellular processes. By examining protein-to-protein interaction networks, researchers may reveal protein complexes, functional units, and disease-related subnetworks. Community detection also finds application in image segmentation (Shi and Malik (1997)). By modeling an image as a graph—with pixels acting as nodes and relationships between pixels forming edges—community detection algorithms can identify pixel clusters corresponding to the same object or image section. This methodology offers several advantages, including the ability to manage large images, incorporate prior information, and capture complex structures. Community detection also plays a pivotal role in improving recommendation systems. It empowers algorithms to deliver more precise predictions of user preferences, informed by item popularity within a community. Additionally, community detection enhances network visualization techniques. By enabling the coloring or labeling of the network, it facilitates a more intuitive understanding of network structure and the relationships among its components.

In summary, community detection is a crucial tool for analyzing and understanding complex systems, as it provides valuable insights into the structure of social networks, biological networks, economic networks, and infrastructure networks. By detecting communities in these networks, researchers and practitioners can understand the relationships between individuals or organizations, how information spreads, and the underlying patterns in these systems.

A hypergraph is an extension of graphs that offers a higher level of complexity. While a graph allows for relationships between pairs of nodes through edges, it may not fully encapsulate the complexity of interactions in some systems. Specifically, it falls short in representing multi-way interactions among more than two nodes simultaneously. A node in a hypergraph represents an individual entity within the system, and the edges in a hypergraph, referred to as hyperedges, extend beyond the binary relationships encapsulated in traditional graph theory. They can connect any number of nodes, providing a mechanism to model complex, multi-way relationships among entities.

A hypergraph $H = (V, E)$ consists of a set V of vertices and a set E of hyperedges. Unlike an edge in a graph, a hyperedge can connect any number of nodes, not just two. In a hypergraph,

as shown in Figure 1.3, the concepts of edges take on a slightly different form. Concretely, in Figure 1.3, the vertex set is defined as $V = \{V_1, V_2, V_3, V_4, V_5, V_6, V_7\}$, and the hyperedge set is denoted as $E = \{e_1, e_2, e_3, e_4\}$. Specifically, hyperedge e_1 forms a connection between vertices V_1, V_2, and V_3; hyperedge e_2 connects vertices V_2 and V_3; hyperedge e_3 forms a link between V_3, V_5, and V_6; and hyperedge e_4 establishes a connection exclusively with V_4. This detailed representation illustrates the versatility of hypergraphs, where hyperedges can connect varying numbers of vertices. The ability to form such complex connections makes hypergraphs an invaluable tool in modeling and analyzing systems where multi-way interactions are inherent. This feature allows for a more accurate representation of multi-way relationships prevalent in many real-world systems. For instance, in social networks, a co-authorship or a collaborative project involves multi-way interactions (Estrada and odriguez-velasquez (2005); Newman (2001); Ouvrard and Marchand-Maillet (2017)) that can be effectively represented as a hyperedge in a hypergraph. Similarly, in a login network (Ghoshdastidar and Dukkipati (2017)), an edge may represent a (user, remote host, login time, logout time) structure that goes beyond pairwise interactions, making hypergraphs a fitting choice for representation. The community structure within a hypergraph is defined similarly to that within a graph, as demonstrated in Figure 1.2. Specifically, Figure 1.4 presents a hypergraph without a distinct community structure, while Figure 1.5 displays a hypergraph with a clear two-community structure. These illustrations effectively represent the range of possible community configurations in a hypergraph.

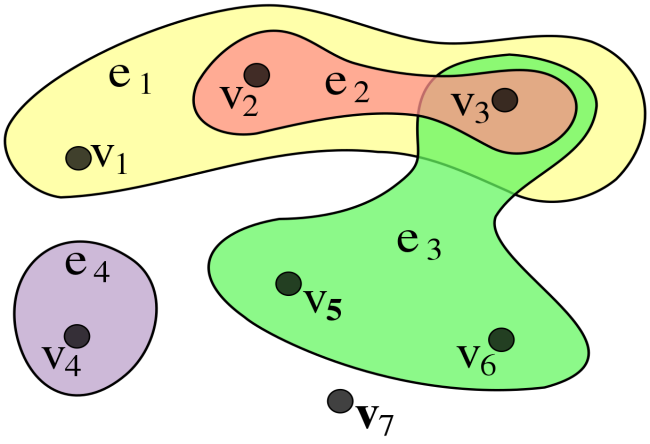

Figure 1.3. An undirected hypergraph

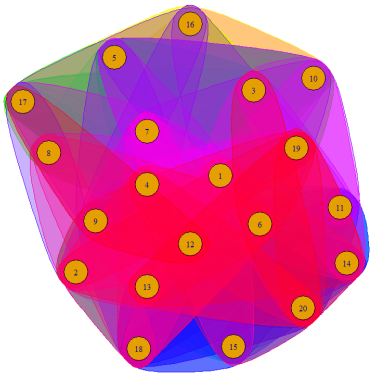

Figure 1.4. A hypergraph without communities

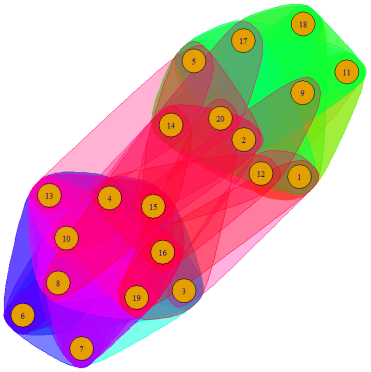

Figure 1.5. A two-community hypergraph

Existing studies on community detection in graph or hypergraph can be classified into two categories: (1) those that derive an information-theoretic threshold to recover the community structure (Abbe et al. (2016); Mossel et al. (2015); Mossel et. al (2017); Chien et al. (2018); Dhara et al. (2021); Hajek et al. (2018); Yuan and Shang (2021)); and (2) those that devise efficient algorithms to recover the community structure (Ghoshdastidar and Dukkipati (2014, 2017); Luo and Zhang (2020); Liu et al. (2015); Ke et al. (2020); Yuan and Qu (2021); Ahn et al. (2018, 2019); Hajek et al. (2016); Gao et al. (2016); Weng and Feng (2021); Zhen and Wang (2021); Lei and Rinaldo (2015); Jin (2015)); see Abbe (2018); Bi et al. (2021) for additional references. The aforementioned methods all apply to networks without missing values.

1.2. Missing Values in Networks

In practice, network data may not always be fully observed. For instance, in a social network, missingness can occur due to the non-responsiveness of actors, leading to absent ties (Huisman, M. (2009); Gile and Handcock (2016)). In the field of medical imaging, specifically in MRI networks, missingness can arise from the high costs associated with PET scanning (Liu et al. (2018)). In the following sections, we introduce two specific hypergraphs characterized by missing values. We emphasize their unique features and detail the methodologies applied in their construction.

1.2.1. Missing Data in Medical Field

Hypergraphs have emerged as a valuable tool in modeling medical data, largely due to their inherent capacity to portray polyadic interactions amongst entities. This quality makes them apt

for the exploration of intricate systems wherein entities perform unique roles. Cai et al. (2022) proposed a method rooted in hypergraph structure to extract patient representations from electronic health records, where individual nodes stand for medical codes, and the hyperedges represent patients. Furthermore, Dai, Q. and Gao, Y. (2023) unveiled four archetypical applications of hypergraph computation in medical and biological scenarios, wherein, during computer-aided diagnosis, nodes symbolize medical images or patches and hyperedges signify feature similarity or high-order topological links.

The development of hypergraphs can be facilitated via straightforward algorithms like the nearest neighbor method. It offers computational efficiency and a certain level of data density, by creating a hyperedge from each vertex to its k nearest neighbors, based on attribute similarity. The total count of hyperedges in the resulting hypergraph hinges on the selection of neighbor size and vertex quantity, as illustrated in Figure 1.6.

In a medical data context, these attributes could encompass various patient-centric factors like PET, CSF, and MRI (structural magnetic resonance imaging) values. Nevertheless, in certain situations, all attributes might not be accessible for every patient. Such incompleteness could be attributed to numerous reasons, such as the elevated cost of PET scans, inferior data quality, or patient attrition during the testing process, which inhibits further data collection, leading to missing values. Consequently, the resulting hypergraphs are incomplete, with certain nodes lacking specific attributes. Notably, a hypergraph regularized transductive learning method has been applied to incomplete multi-modality data in the ADNI database for automatic diagnosis of brain diseases (Liu et al. (2018)), underscoring the feasibility and potential of handling incomplete hypergraphs in medical data analysis.

Later sections will elaborate on the approach to handling these missing values within the context of censored hypergraph algorithms.

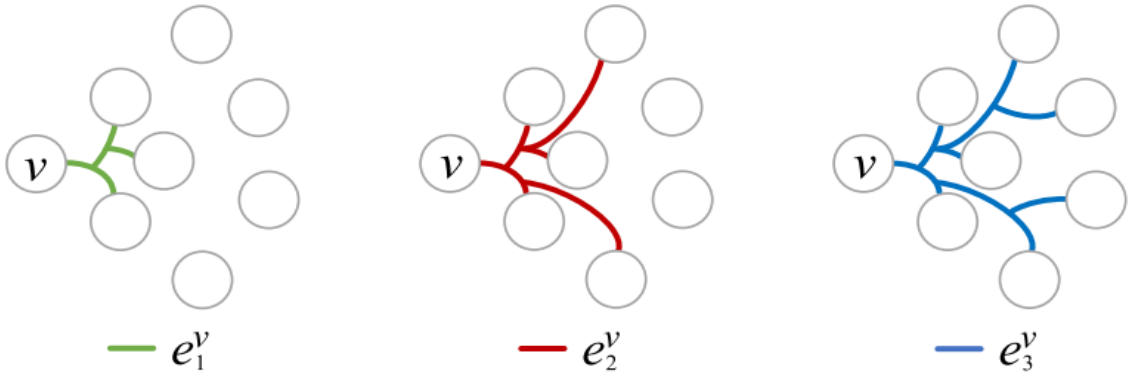

Figure 1.6. Hypergraph construction for medical data. We construct the hypergraph using the nearest-neighbor algorithm based on the attributes that a patient has.

1.2.2. Missing Data in Engineering Field

Apart from the medical domain, engineering researchers also grapple with the issue of missing data. Their solution often involves converting the raw data into hypergraphs, as noted by Hu and Shi (2015). The concept of the neighborhood model was introduced by Lin in 1988 (Lin, T. Y. (1998)). Subsequently, Hu et al. (2008) explored the properties of neighborhood approximation spaces and proposed a neighborhood-based rough set model. Later, Hu et al. (2014) utilized the neighborhood rough set to classify data by calculating the neighborhood threshold for samples based on the Minkowski distance, encompassing all the attribute values of the calculated sample.

This approach diverges from the previous example where hypergraphs were created using the nearest neighbor algorithm. Instead, it constructs hypergraphs based on attribute similarity between samples, coupled with the theory of neighborhood rough sets. When a threshold is defined, and the attribute similarity between two samples falls beneath this threshold, they are grouped within the same hyperedge, leading to the formation of a hypergraph. If certain attributes contain missing values, the associated hyperedges are either absent or missing. Figure 1.7 visually represents this process of hypergraph construction in engineering data.

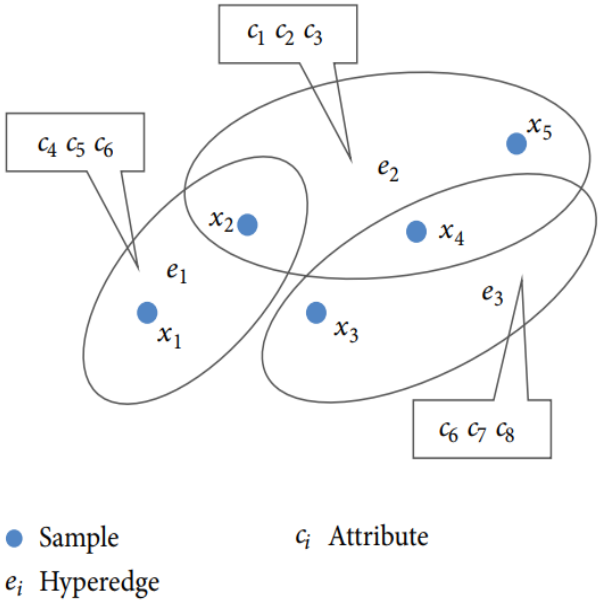

Figure 1.7. Hypergraph construction for engineering data. We construct the hypergraph by calculating the attribute similarity based on the attributes that a node in the engineering data has.

1.3. Community Detection in Censored Networks

Missing values have non-negligible effects on the structural properties of a network (Huisman, M. (2009); Smith et al. (2018)), and most existing algorithms for community detection apply to uncensored networks. Thus, a natural question is how to recover communities in a censored network. Researchers can deal with missing values, typically present in data gathered for empirical research, in various methods. The simplest course of action is simply ignoring the missing data and focusing on the observed replies. However, some researchers (Little and Rubin (2019)) believe that ignoring the missing values and only analyzing the observed data will result in a severe loss of information and a decrease in statistical power. Except for the deletion method (we delete all missing values from the graph or hypergraph), weighting procedures, and model-based procedures, some imputation methods are also proposed to deal with missing data. For instance, in Huisman, M. (2009); Smith et al. (2018)'s work, they illustrated several simple imputation methods.

One of the interesting research topics is to study the effect of missing values on the performance of community detection algorithms. To the best of our knowledge, Abbe et al. (2014) was the first to theoretically examine community detection in a censored graph, obtaining an information-theoretic threshold for the exact recovery of communities. Recently, Dhara et al. (2021) introduced

a censored stochastic block model for censored community detection, specifically applicable to scenarios with two communities. In their model, most of the edges are missing, with only a small fraction of potential edges being observed. An information-theoretic threshold is established in their study, if this threshold is not met, the successful recovery of communities by any algorithm is impossible. They propose a spectral algorithm to recover communities when the threshold is met, they also illustrate the limitations of this spectral algorithm in the context of an asymmetric case, which is characterized by differing connection probabilities within the two communities.

Hypergraph learning with missing values has recently attracted much attention. Hu and Shi (2015) introduced a unique classification algorithm tailored to incomplete information systems. Their approach is founded on the integration of a hypergraph model and rough set theory. Liu et al. (2017, 2018) employed a multi-hypergraph to represent higher-order relationships among subjects in the Alzheimer's Disease Neuroimaging Initiative (ADNI) database. The purpose of this representation was to segregate the subjects into distinct groups based on the availability of their respective data modalities. Their approaches demonstrate how hypergraphs can be effectively utilized to manage and analyze missing data, providing a basis for our subsequent exploration.

In this study, we are interested in detecting communities in censored hypergraphs. It is not immediately clear how the results obtained by (Dhara et al. (2021)) changes in the case of a censored hypergraph. Our contributions to the literature are summarized as follows. We derive an information-theoretic threshold for the exact recovery of a community structure in a censored uniform hypergraph. Interestingly, the threshold is larger, in general, than that in the graph case. In this sense, community detection in a censored hypergraph is more difficult than it is in the case of a censored graph. In addition, we propose a polynomial-time algorithm that can exactly recover the community structure up to the information-theoretic threshold. The proposed algorithm consists of a spectral algorithm plus a refinement step. We also study whether a single spectral algorithm without refinement can achieve the threshold as the censored graph case (Dhara et al. (2021)). To this end, we study the semi-definite relaxation algorithm and provide a sufficient condition for the algorithm to achieve exact recovery. Moreover, we propose several methods to impute the missing values in hypergraphs and use simulation to evaluate their performance.

1.4. Organization of the Content

The rest of this paper is structured as follows. Section 2 provides a summary of the key results of the censored hypergraph stochastic block model. The proofs for these theorems are given in Section 3. Section 4 shares the results of numerical simulations conducted for our proposed clustering algorithms, evaluates the performance of the suggested imputation methods and presents the simulation outcomes for community detection in multi-community hypergraphs. In Section 5, we demonstrate the real-world applicability of our proposed algorithm by using it on an actual dataset. Lastly, we give an overall summary of this study in section 6, stating our contributions and conclusions.

2. MAIN RESULTS

In this section, we delineate an information-theoretic threshold for the exact recovery in the context of a Censored Hypergraph Stochastic Block Model (CHSBM). We subsequently introduce the Efficient Spectral algorithm, which is a two-stage method designed for the exact recovery of the community structure. Additionally, we propose a one-stage algorithm based on Semi-Definite Programming. To complement these, we also present several imputation methods, each of which leverages distinct network properties. This multifaceted approach provides a comprehensive examination of the tools and techniques integral to the analysis and manipulation of censored hypergraphs.

2.1. The Censored Hypergraph Stochastic Block Model

For a positive integer n, we define a set of nodes $\mathcal{V} = \{1, 2, \ldots, n\}$ and a set of subsets of $\mathcal{V}$ denoted by $\mathcal{E}$. An undirected m-uniform hypergraph $\mathcal{H}_m = (\mathcal{V}, \mathcal{E})$ is characterized by the property that each hyperedge $e \in \mathcal{E}$ contains exactly m distinct nodes. We represent $\mathcal{H}_m$ as a symmetric m-dimensional array $A = (A_{i_1,\ldots,i_m}) \in \{0,1\}^{\otimes n^m}$, where $A_{i_1 i_2 \ldots i_m} = 1$ if $\{i_1, i_2, \ldots, i_m\}$ forms a hyperedge, and $A_{i_1 i_2 \ldots i_m} = 0$ otherwise. It should be noted that $A_{i_1 i_2 \ldots i_m} = A_{j_1 j_2 \ldots j_m}$ when $\{i_1, i_2, \ldots, i_m\} = \{j_1, j_2, \ldots, j_m\}$. In this paper, we exclude self-loops, meaning that $A_{i_1 i_2 \ldots i_m} = 0$ when $|\{i_1, i_2, \ldots, i_m\}| < m$. When $m = 2$, $\mathcal{H}_2$ represents a conventional graph widely employed in community detection problems (Abbe (2018)). We specifically focus on the hypergraph derived from the Censored m-uniform Hypergraph Stochastic Block Model (CHSBM) denoted as $\mathcal{H}_m(n, p, q, \alpha)$ throughout this paper.

Definition 2.1.1 (Censored m-uniform Hypergraph Stochastic Block Model (CHSBM))**.** *Each node $i \in \mathcal{V}$ is randomly and independently assigned a label σ_i with*

$$\mathbb{P}(\sigma_i = +1) = \mathbb{P}(\sigma_i = -1) = \frac{1}{2}.$$

Let $\sigma = (\sigma_1, \ldots, \sigma_n)^T$ be a column vector of labels, $I_+(\sigma) = \{i | \sigma_i = +1\}$ and $I_-(\sigma) = \{i | \sigma_i = -1\}$. The nodes in $I_+(\sigma)$ and $I_-(\sigma)$ constitute two communities. The distinct nodes $i_1, i_2, \ldots, i_m$ form a hyperedge with probability p if $\{i_1, i_2, \ldots, i_m\}$ is a subset of $I_+(\sigma)$ or $I_-(\sigma)$ and q otherwise. Each hyperedge status is revealed independently with probability α. A hyperedge of the resulting hypergraph

takes value in $\{1, 0, *\}$*, where* $*$ *means a hyperedge is missing (the hyperedge status is not revealed). This model is denoted as* $\mathcal{H}_m(n, p, q, \alpha)$.

In the censored hypergraph model $\mathcal{H}_m(n, p, q, \alpha)$ with $\alpha < 1$, each hyperedge can have one of three possible states: 1 (present), 0 (absent), or $*$ (missing). When $\alpha = 1$, the hypergraph is uncensored, and $\mathcal{H}_m(n, p, q, 1)$ corresponds to the standard hypergraph stochastic block model (Ghoshdastidar and Dukkipati (2014, 2017); Chien et al. (2018); Kim et al. (2018); Ke et al. (2020); Yuan and Shang (2021)). The censored stochastic block model $CSBM(p, q, \alpha)$ studied in (Dhara et al. (2021)) corresponds to $\mathcal{H}_2(n, p, q, \alpha)$. In this paper, we assume fixed constants $p, q \in (0, 1)$ where $p > q$, and $\alpha = \frac{t \log n}{n^{m-1}}$ for a constant $t > 0$. We consider the order of α as $\frac{\log n}{n^{m-1}}$ because it is the smallest order that allows for exact recovery (see Theorem 2.1 and Theorem 2.2).

For the in-community and cross-community probabilities aforementioned, assume we have 6 nodes in a 2-uniform hypergraph, the first 3 belong to one community, and the rest belong to the other community. Then the probability matrix will be:

$$P_{ij} = \begin{pmatrix} 0 & 0.4 & 0.4 & 0.1 & 0.1 & 0.1 \\ 0.4 & 0 & 0.4 & 0.1 & 0.1 & 0.1 \\ 0.4 & 0.4 & 0 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0 & 0.4 & 0.4 \\ 0.1 & 0.1 & 0.1 & 0.4 & 0 & 0.4 \\ 0.1 & 0.1 & 0.1 & 0.4 & 0.4 & 0 \end{pmatrix}$$

where $p = 0.4$ and $q = 0.1$.

Given a hypergraph A generated from $\mathcal{H}_m(n, p, q, \alpha)$, community detection refers to the problem of recovering the unknown true label vector σ, or equivalently, identifying the sets $I_+(\sigma)$ and $I_-(\sigma)$. We say an estimator $\hat{\sigma}$ is an exact recovery of σ, $\hat{\sigma}$ exactly recovers σ, or $\hat{\sigma}$ achieves an exact recovery if

$$\mathbb{P}(\exists s \in \{\pm 1\} : \hat{\sigma} = s\sigma) = 1 - o(1).$$

That is, the estimator $\hat{\sigma}$ is equal to σ or $-\sigma$ with probability $1 - o(1)$. If there exists an estimator $\hat{\sigma}$ that exactly recovers σ, we say an exact recovery is possible. Otherwise, we say that an exact recovery is not possible.

2.2. Sharp Threshold for Exact Recovery

In this subsection, we derive a sharp phase transition threshold for exact recovery. The first result specifies a sufficient condition for the impossibility of exact recovery.

Theorem 2.1. *For each fixed integer $m \geq 2$, if $t < I_m(p,q)$, then $\mathbb{P}(\hat{\sigma} = \sigma) = o(1)$ for any estimator $\hat{\sigma}$. Here $I_m(p,q)$ is defined as*

$$I_m(p,q) = \frac{2^{m-1}(m-1)!}{(\sqrt{p}-\sqrt{q})^2 + (\sqrt{1-p}-\sqrt{1-q})^2}. \tag{2.1}$$

Theorem 2.1 establishes that if $t < I_m(p,q)$, no estimator can achieve exact recovery of the true labels. In the case of $m = 2$, $I_2(p,q)$ corresponds to $t_c(p,q)$ as defined in (Dhara et al. (2021)). Our result can be viewed as a nontrivial extension of Theorem 2.1 in (Dhara et al. (2021)). Interestingly, when p and q are fixed, the region where $t < I_2(p,q)$ is smaller compared to the region where $t < I_m(p,q)$ for $m \geq 3$. A similar phenomenon is observed in the exact recovery of communities within an uncensored hypergraph stochastic block model (Kim et al. (2018)). However, it differs notably from the hypothesis testing scenario for communities. For instance, Yuan and Shang (2021) derived the sharp boundary for testing the presence of a dense subhypergraph. When the number of nodes in the dense subhypergraph is not excessively small, the region in which any test becomes asymptotically powerless for $m = 2$ is larger than that for $m \geq 3$.

The next result shows that the threshold $I_m(p,q)$ is actually sharp for exact recovery.

Theorem 2.2. *For each fixed integer $m \geq 2$, if $t > I_m(p,q)$, with $I_m(p,q)$ defined in (2.1), then the MLE exactly recovers the true label with probability $1 - o(1)$.*

According to Theorem 2.2, when $t > I_m(p,q)$, we can use the Maximum Likelihood Estimation (MLE) to exactly recover the true labels. By combining Theorem 2.1 and Theorem 2.2, we establish the sharp boundary $t = I_m(p,q)$ for exact recovery, which forms a surface in $\mathbb{R}^3$. To provide a visual representation, we illustrate the regions $t > I_m(p,q)$ and $t < I_m(p,q)$ for $q = 0.2$ and $m = 2, 3$ in Figure 2.1. The gray region represents $t < I_m(p, 0.2)$ where exact recovery is impossible, while the white region corresponds to $t > I_m(p, 0.2)$ where exact recovery is possible. It is evident that the white region for $m = 3$ is smaller than that for $m = 2$. Thus, as m increases, exact recovery becomes more challenging.

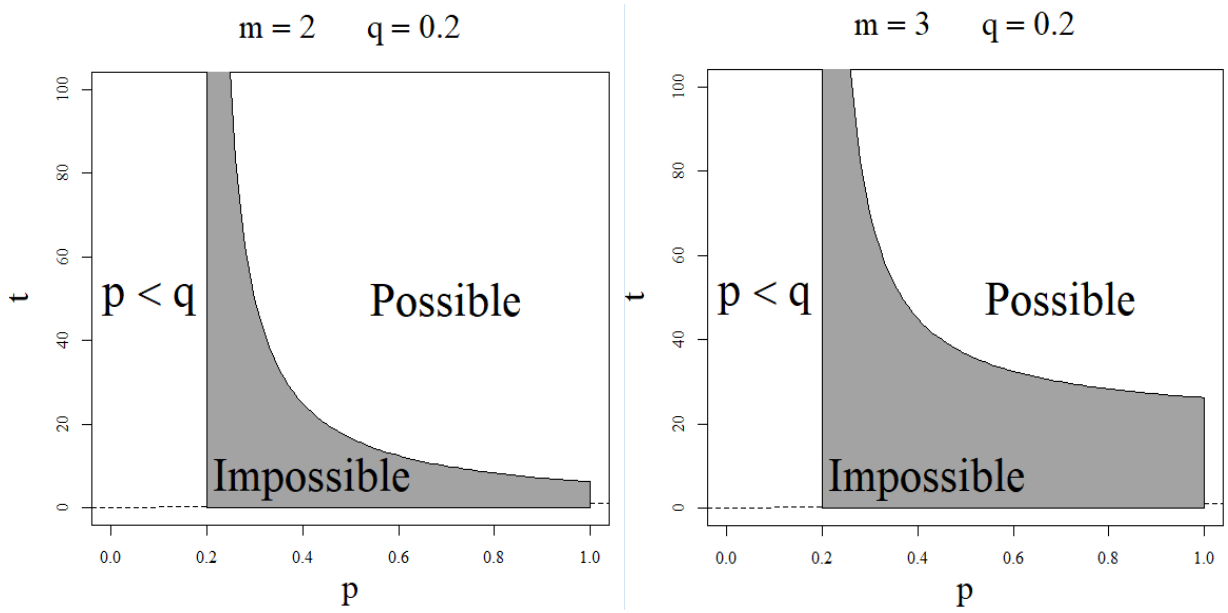

Figure 2.1. Regions for exact recovery with $m = 2, 3$ and $q = 0.2$. Gray: exact recovery is impossible. White: exact recovery is possible.

2.3. Efficient Spectral Algorithm for Exact Recovery

Due to the non-polynomial time complexity of Maximum Likelihood Estimation (MLE) for exact recovery, we propose an efficient algorithm to reconstruct two communities up to the information-theoretic threshold. The algorithm starts with a random splitting of the hypergraph A into two parts. Then, we applied a spectral algorithm to the first part, followed by a refinement based on the second part. We describe the algorithm in the following three steps.

In the first step, we randomly split the hypergraph A into two parts. Denote $M_m = \{(i_1, i_2, \ldots, i_m) \mid 1 \le i_1 < \cdots < i_m \le n\}$. Let S_1 be a random subset of M_m obtained by including each element of M_m in S_1 with probability $\frac{\log\log n}{\log n}$. Let S_2 be the compliment of S_1 in M_m, that is, $S_2 = M_m - S_1$. Define a hypergraph $\tilde{A}$ as

$$\tilde{A}_{i_1 i_2 \ldots i_m} = \begin{cases} \mathbb{1}[A_{i_1 i_2 \ldots i_m} = 1], & \{i_1, i_2, \ldots, i_m\} \in S_1, \\ 0, & \text{otherwise.} \end{cases}$$

Here, $\mathbb{1}[E]$ is the indicator function of event E. Define hypergraph $\bar{A}$ as

$$\bar{A}_{i_1 i_2 \ldots i_m} = \begin{cases} A_{i_1 i_2 \ldots i_m}, & \{i_1, i_2, \ldots, i_m\} \in S_2, \\ *, & \text{otherwise.} \end{cases}$$

Then, we randomly divided hypergraph A into two independent hypergraphs $\tilde{A}$ and $\bar{A}$.

In the second step, we apply the weak recovery algorithm HSC in (Ahn et al. (2018)) to $\tilde{A}$. The HSC algorithm converts a hypergraph $\tilde{A}$ to an $n \times n$ similarity matrix B using $B_{ij} = \sum_{1 \le i_3 < i_4 < \cdots < i_m \le n} \tilde{A}_{iji_3 i_4 \ldots i_m}$, and then applies geometric two-clustering to the top two eigenvectors of B to output the communities $\tilde{I}_+(\sigma)$ and $\tilde{I}_-(\sigma)$. The sampling probability $\frac{\log\log n}{\log n}$ in the first step ensures that the hyperedge probability of $\tilde{A}$ has order $\frac{\log\log n}{\log n}\alpha = \frac{t \log\log n}{n^{m-1}}$ (Here, the $\log\log n$ factor can be replaced by any a_n with $a_n \to \infty$). According to Theorem 1 of Ahn et al. (2018), $n - o(n)$ of the nodes are correctly labeled by the HSC algorithm with probability $1 - o(1)$.

The last step is to refine the communities $\tilde{I}_+(\sigma)$ and $\tilde{I}_-(\sigma)$ based on $\bar{A}$. For a set $S \subset [n]$, define $e(i, S)$ as

$$e(i, S) = \sum_{\substack{i_2, \ldots, i_m \in S \setminus \{i\} \\ i_2 < \cdots < i_m}} \left(\log\left(\frac{p}{q}\right) \mathbb{1}[\bar{A}_{i i_2 \ldots i_m} = 1] + \log\left(\frac{1-p}{1-q}\right) \mathbb{1}[\bar{A}_{i i_2 \ldots i_m} = 0] \right).$$

For each node $i \in \tilde{I}_+(\sigma)$, flip the label of i if

$$e(i, \tilde{I}_+(\sigma)) < e(i, \tilde{I}_-(\sigma)).$$

For each node $j \in \tilde{I}_-(\sigma)$, flip the label of j if

$$e(j, \tilde{I}_-(\sigma)) < e(j, \tilde{I}_+(\sigma)).$$

Let $\hat{I}_+(\sigma)$ and $\hat{I}_-(\sigma)$ be the resulting communities. If $|\hat{I}_+(\sigma)| \neq |\tilde{I}_+(\sigma)|$, output $\tilde{I}_+(\sigma)$ and $\tilde{I}_-(\sigma)$; otherwise output $\hat{I}_+(\sigma)$ and $\hat{I}_-(\sigma)$.

The above algorithm is summarized in Algorithm 1.

Theorem 2.3. *For each fixed integer $m \ge 2$, if $t > I_m(p, q)$, with $I_m(p, q)$ defined in (2.1), then Algorithm 1 exactly recovers the true label with probability $1 - o(1)$.*

The time complexity of Algorithm 1 is bounded by $O(n^m)$. This complexity analysis applies to the random splitting in Step 1 as well as the refinement in Step 3, both of which have a time complexity of at most $O(n^m)$. In Step 2, the weak recovery algorithm HSC proposed by Ahn et al. (2018) also exhibits a time complexity of $O(n^m)$, as documented in the comments below Remark 1 of their publication. Consequently, Theorem 2.3 establishes that an algorithm with polynomial time complexity can achieve the information-theoretic threshold.

Algorithm 1: Spectral algorithm plus refinement for exact recovery

---

Input: A censored m-uniform hypergraph A generated from $\mathcal{H}_m(n, p, q, \alpha)$.

Step 1: random splitting.

Randomly select elements in $M_m = \{(i_1, i_2, \ldots, i_m) \mid 1 \le i_1 < \cdots < i_m \le n\}$ with probability $\frac{\log\log n}{\log n}$ to form a subset $S_1 \subset M_m$ and let $S_2 = M_m - S_1$. Construct the hypergraph $\tilde{A}$ as $\tilde{A}_{i_1 i_2 \ldots i_m} = \mathbb{1}[A_{i_1 i_2 \ldots i_m} = 1]$, if $i_1, i_2, \ldots, i_m \in S_1$ and $\tilde{A}_{i_1 i_2 \ldots i_m} = 0$ otherwise. Construct the hypergraph $\bar{A}$ as $\bar{A}_{i_1 i_2 \ldots i_m} = A_{i_1 i_2 \ldots i_m}$ if $i_1, i_2, \ldots, i_m \in S_2$ and $\bar{A}_{i_1 i_2 \ldots i_m} = *$ otherwise.

Step 2: spectral algorithm.

Apply the weak recovery algorithm HSC in Ahn et al. (2018) to $\tilde{A}$, and denote the community output as $\tilde{I}_+(\sigma)$, $\tilde{I}_-(\sigma)$.

Step 3: refinement.

Flip the label of $i \in \tilde{I}_+(\sigma)$ if $e(i, \tilde{I}_+(\sigma)) < e(i, \tilde{I}_-(\sigma))$.

Flip the label of $j \in \tilde{I}_-(\sigma)$ if $e(j, \tilde{I}_-(\sigma)) < e(j, \tilde{I}_+(\sigma))$.

Let $\hat{I}_+(\sigma)$ and $\hat{I}_-(\sigma)$ be the resulting communities.

Output: If $|\hat{I}_+(\sigma)| \neq |\tilde{I}_+(\sigma)|$, output $\tilde{I}_+(\sigma)$ and $\tilde{I}_-(\sigma)$;
otherwise output $\hat{I}_+(\sigma)$ and $\hat{I}_-(\sigma)$.

2.4. Semi-Definite Relaxation Algorithm

In subsection 2.3, we demonstrate that by employing a spectral algorithm with a refinement step, we can achieve exact recovery. However, it is intriguing to investigate whether a single spectral algorithm, without a subsequent refinement step, can reach the information-theoretic threshold. In the case of graphs ($m = 2$), the answer is affirmative, as both the semi-definite relaxation algorithm

and the spectral algorithm have been shown to succeed without the need for a refinement step (Hajek et al. (2016); Dhara et al. (2021)). Nevertheless, for hypergraphs ($m \geq 3$), regardless of whether they are censored or uncensored, this question remains an open problem. In this subsection, we delve into the semi-definite relaxation algorithm and conduct an analysis of its performance. To accomplish this, we introduce a new hypergraph derived from the given hypergraph A and transform it into a weighted graph. Subsequently, we demonstrate that by applying the semi-definite relaxation algorithm to the weighted graph, we can achieve exact recovery.

Define the hypergraph $\widetilde{A}$ based on A as

$$\widetilde{A}_{i_1 i_2 \ldots i_m} = \mathbb{1}[A_{i_1 i_2 \ldots i_m} = 1],$$

and $\widetilde{A}_{i_1 i_2 \ldots i_m} = 0$ if $|\{i_1, i_2, \ldots, i_m\}| \leq m-1$. Each hyperedge $\widetilde{A}_{i_1 i_2 \ldots i_m}$ takes value in $\{1, 0\}$. The hypergraph $\widetilde{A}$ shares the same community structure as that of A, because

$$\mathbb{E}(\widetilde{A}_{i_1 i_2 \ldots i_m}) = \begin{cases} p\alpha, & \{i_1, i_2, \ldots, i_m\} \subset I_+(\sigma) \text{ or } I_-(\sigma); \\ q\alpha, & \text{otherwise.} \end{cases}$$

Next, we construct a weighted graph $G = [G_{ij}]$ based on $\widetilde{A}$ by using

$$G_{ij} = \sum_{1 \leq i_3 < \cdots < i_m \leq n} \widetilde{A}_{iji_3 \ldots i_m}.$$

Define the semi-definite program problem (SDP) as

$$\begin{aligned} \max_{Y} \quad & \langle G, Y \rangle \\ \text{s.t.} \quad & Y \succeq 0 \\ & \langle Y, J \rangle = 0 \\ & Y_{ii} = 1, \ i \in [n], \end{aligned} \tag{2.2}$$

where J is an $n \times n$ all-one matrix. Suppose σ is the true label, and denote $Y_0 = \sigma\sigma^T$. Let $\widehat{Y}$ be the solution to the SDP (2.2). The following result provides a sufficient condition under which $\widehat{Y}$ is an exact recovery of Y_0.

Theorem 2.4. *For each fixed integer $m \geq 2$, let*

$$J_m(p,q) = \frac{2^{m+2}(m-2)!\,[mp-(m-2^m)q]}{(p-q)^2}.$$

If $t > J_m(p,q)$, then $\mathbb{P}(\widehat{Y} = Y) = 1 - o(1)$, where $Y = \sigma\sigma^T$, with true label σ.

Note that $J_m(p,q) > I_m(p,q)$, for each $m \geq 2$. When $m = 2$ and the graph is uncensored, $\widehat{Y}$ can exactly recover the true label up to the information-theoretic threshold (Hajek et al. (2016)). However, for $m \geq 3$, it is unclear whether $\widehat{Y}$ succeeds in the range $I_m(p,q) < t < J_m(p,q)$. A similar gap exists in the uncensored hypergraph case (Kim et al. (2018)).

2.5. Imputation Methods

In the work by Huisman, M. (2009), the researchers explored several basic imputation methods in the graph context. These were categorized into four types of imputation procedures: unconditional mean imputation, conditional distribution imputation, conditional mean imputation, and imputation from conditional distributions. The authors provided some examples of simple imputation methods in their paper, which were originally defined in Schafer and Graham (2002) as follows:

1. **Imputing the unconditional mean:** This method imputes the average tie value over all observed ties. For binary networks, this is equal to the network density.

2. **Imputation by reconstruction:** This method uses observed incoming relations of the missing actors to reconstruct the missing part of the network. If both ties in a dyad are missing, additional imputations are necessary, which are performed randomly proportional to the observed density.

3. **Imputation using preferential attachment:** This method assumes that the probability that a missing actor will be connected to another actor is proportional to the indegree of that actor. It preserves the degree distributions and is expected to perform well in social network research.

4. **Hot Deck imputation:** This method uses completely observed donor actors to replace the missing actor or missing ties of an incomplete actor. Actors are matched on a completely observed attribute.

Motivated by the methods in Huisman, M. (2009), we propose several imputation methods to impute missing values in hypergraphs.

2.5.1. Imputation Using Network Density

For hypergraphs, network density refers to the proportion of observed hyperedges to the total possible number of hyperedges. We impute the missing values of censored hyperedges as either one (present) if the density falls above a predetermined threshold or zero (absent), otherwise.

Let's denote:

- E_o as the set of observed hyperedges in a hypergraph,
- V as the set of vertices in a hypergraph,
- $E_{\max}$ as the maximum possible number of hyperedges that could exist among vertices V,
- d as the network density,
- e_{missing} as the missing hyperedge values,
- T as a predetermined threshold.

Then, we can calculate the network density d in both graph and hypergraph as:

$$d = \frac{|E_o|}{E_{\max}}$$

where $|E_o|$ represents the number of observed hyperedges.

We can represent the imputation of missing hyperedge values as:

$$e_{\text{missing}} = \begin{cases} 1 & \text{if } d > T \\ 0 & \text{otherwise} \end{cases}$$

This indicates that missing values of censored hyperedges are imputed as either one (indicating the hyperedge is present) or zero (indicating the hyperedge is absent) depending on whether the network density falls below a predetermined threshold T.

2.5.2. Imputation Using Community Density

We propose a community density imputation methods as follows: we first apply a community detection algorithm to the observed hypergraph, then calculate the in-community and cross-community density. These densities are then used as the occurrence probabilities of Bernoulli distributions $B(p)$. Finally, we impute the missing hyperedges by random numbers drawn from their respective Bernoulli Distributions, creating a new set of hyperedges.

The steps to implement this method are as follows:

1. Apply a community detection algorithm to the observed hypergraph (that is, the missing hyperedges are set to be zero), thereby dividing it into a number of communities $C = \{C_1, C_2, \ldots, C_r\}$.

2. For each community C_i and each pair of communities (C_i, C_j), calculate the in-community density d_{C_i} and the cross-community density $d_{C_iC_j}$ as the ratios of the number of observed hyperedges to the maximum possible number of hyperedges within or between the respective communities.

3. Use these calculated densities to form multiple Bernoulli distributions, $B(p)$, where $p = d_{C_i}$ or $p = d_{C_iC_j}$.

4. Impute the missing hyperedges by drawing from the Bernoulli distributions corresponding to each missing hyperedge's community or pair of communities. This forms a new set of hyperedges.

2.5.3. Imputation Using Degree

We propose a new degree-based imputation method. The method uses the degree of each vertex as an indicator of comparison. The degree of a node v is defined as $deg(v) = \sum_{e \in \varepsilon} H_{ve}$, which is simply the sum of the entries in the vth row of the incidence matrix H.

To impute the missing data, we first set the status of missing hyperedges $A_{i_1,i_2,\ldots,i_m}$, where $i_1, i_2, \ldots, i_m$ are distinct nodes, to 0, indicating the absence of connections between them. Next, we calculate the degree of all vertices. We then set a threshold to reveal the status of the missing hyperedges. If a missing hyperedge contains nodes with degrees greater than the average node

degree, we set the corresponding missing hyperedge to present (represented by 1 in the incidence matrix); otherwise, it is set to absent (represented by 0 in the incidence matrix). These steps complete the imputation process.

The implementation of this imputation method proceeds as follows:

1. Initially set the status of missing hyperedges $A_{i_1,i_2,\ldots,i_m}$, where $i_1, i_2, \ldots, i_m$ are distinct nodes, to 0, indicating the absence of hyperedges between these nodes.

2. Calculate the degree of all vertices in the hypergraph.

3. Determine a threshold based on the average node degree.

4. For each missing hyperedge, if it contains nodes with degrees exceeding the threshold, then set the status of this hyperedge to 1 in the incidence matrix, indicating its presence. Otherwise, set it to 0, indicating its absence.

3. PROOF OF THEOREMS

3.1. Proof of Theorem 2.1

In this section, we prove Theorem 2.1.

Proof of Theorem 2.1 : Let $l(\sigma)$ denote the log-likelihood function associated with a label σ. It is important to note that, according to Definition 2.1, the true label vector σ is uniformly and independently selected from the set $S = \{\pm 1\}^n$. Proposition 4.1 in Dhara et al. (2021) states that if there exist labels η_t $(1 \leq t \leq k_n)$, where $k_n \to \infty$, such that $l(\eta_1) = l(\eta_2) = \cdots = l(\eta_{k_n}) = l(\sigma)$, then the MLE fails to achieve exact recovery of the true label with a probability of $1 - o(1)$. In our proof, we construct labels η_t $(1 \leq t \leq k_n)$ under the condition $t < I_m(p, q)$, where $k_n \to \infty$.

First, we write down the explicit expression of the likelihood function. Note that for distinct nodes $i_1, i_2, \ldots, i_m$, we have

$$A_{i_1 i_2 \ldots i_m} = \begin{cases} 1 & , \\ 0 & , \\ * & . \end{cases}$$

For convenience, let $\mathbb{1}[E]$ be the indicator function of event E and

$$\mathbb{1}_{i_1 i_2 \ldots i_m}(\sigma) = \mathbb{1}[\sigma_{i_1} = \sigma_{i_2} = \cdots = \sigma_{i_m}].$$

Then, the likelihood function for σ given an observation of hypergraph A from $\mathcal{H}_m(n,p,q,\alpha)$ is

$$\begin{aligned}
L &= \prod_{1\le i_1<\cdots<i_m\le n} (p\alpha)^{\mathbb{1}[A_{i_1i_2\ldots i_m}=1]\mathbb{1}_{i_1i_2\ldots i_m}(\sigma)}[\alpha(1-p)]^{\mathbb{1}[A_{i_1i_2\ldots i_m}=0]\mathbb{1}_{i_1i_2\ldots i_m}(\sigma)} \\
&\quad \times (q\alpha)^{\mathbb{1}[A_{i_1i_2\ldots i_m}=1](1-\mathbb{1}_{i_1i_2\ldots i_m}(\sigma))}[\alpha(1-q)]^{\mathbb{1}[A_{i_1i_2\ldots i_m}=0](1-\mathbb{1}_{i_1i_2\ldots i_m}(\sigma))}(1-\alpha)^{\mathbb{1}[A_{i_1i_2\ldots i_m}=*]} \\
&= \prod_{1\le i_1<\cdots<i_m\le n} (1-\alpha)^{\mathbb{1}[A_{i_1i_2\ldots i_m}=*]}(q\alpha)^{\mathbb{1}[A_{i_1i_2\ldots i_m}=1]}\left(\frac{p}{q}\right)^{\mathbb{1}[A_{i_1i_2\ldots i_m}=1]\mathbb{1}_{i_1i_2\ldots i_m}(\sigma)} \\
&\quad \times[\alpha(1-q)]^{\mathbb{1}[A_{i_1i_2\ldots i_m}=0]}\left(\frac{1-p}{1-q}\right)^{\mathbb{1}[A_{i_1i_2\ldots i_m}=0]\mathbb{1}_{i_1i_2\ldots i_m}(\sigma)} \\
&= \prod_{1\le i_1<\cdots<i_m\le n} (1-\alpha)^{\mathbb{1}[A_{i_1i_2\ldots i_m}=*]}(q\alpha)^{\mathbb{1}[A_{i_1i_2\ldots i_m}=1]}[\alpha(1-q)]^{\mathbb{1}[A_{i_1i_2\ldots i_m}=0]} \\
&\quad \times \prod_{1\le i_1<\cdots<i_m\le n}\left(\frac{p}{q}\right)^{\mathbb{1}[A_{i_1i_2\ldots i_m}=1]\mathbb{1}_{i_1i_2\ldots i_m}(\sigma)}\left(\frac{1-p}{1-q}\right)^{\mathbb{1}[A_{i_1i_2\ldots i_m}=0]\mathbb{1}_{i_1i_2\ldots i_m}(\sigma)}.
\end{aligned}$$

We obtain the MLE by maximizing L with respect to σ. The first product factor of L does not involve σ. Hence, we need only maximize the second product factor of L to obtain the MLE. Denote

$$l(\sigma) = \sum_{1\le i_1<\cdots<i_m\le n}\left[\log\left(\frac{p}{q}\right)\mathbb{1}[A_{i_1i_2\ldots i_m}=1]\mathbb{1}_{i_1i_2\ldots i_m}(\sigma)+\log\left(\frac{1-p}{1-q}\right)\mathbb{1}[A_{i_1i_2\ldots i_m}=0]\mathbb{1}_{i_1i_2\ldots i_m}(\sigma)\right].$$

The log-likelihood function is equal to

$$\log L = R_n + l(\sigma), \tag{3.1}$$

where R_n is independent of σ.

Below, we construct labels η_t $(1\le t\le k_n)$ with $k_n\to\infty$ under the condition $t<I_m(p,q)$. Because R_n is independent of σ, we need only focus on $l(\sigma)$.

Note that

$$\begin{aligned}
l(\sigma) &= \left[\log\left(\frac{p}{q}\right)\mathbb{1}[A_{i_1\ldots i_m}=1]+\log\left(\frac{1-p}{1-q}\right)\mathbb{1}[A_{i_1\ldots i_m}=0]\right]\mathbb{1}[\sigma_{i_1}=\cdots=\sigma_{i_m}=+1] \\
&\quad +\left[\log\left(\frac{p}{q}\right)\mathbb{1}[A_{i_1\ldots i_m}=1]+\log\left(\frac{1-p}{1-q}\right)\mathbb{1}[A_{i_1\ldots i_m}=0]\right]\mathbb{1}[\sigma_{i_1}=\cdots=\sigma_{i_m}=-1].
\end{aligned}$$

Let's assume that $i_0\in I_+(\sigma)$ has exactly m_1 present hyperedges and m_2 absent hyperedges in $I_+(\sigma)$, and it also has exactly m_1 present hyperedges and m_2 absent hyperedges in $I_-(\sigma)$. Similarly,

let's consider $j_0 \in I_-(\sigma)$, which has exactly m_1 present hyperedges and m_2 absent hyperedges in $I_+(\sigma)$, as well as m_1 present hyperedges and m_2 absent hyperedges in $I_-(\sigma)$. It is worth noting that flipping the labels of i_0 and j_0 does not alter the value of $l(\sigma)$. Let $\tilde{\sigma}$ represent the label configuration obtained by flipping the labels of i_0 and j_0. It can be verified that $l(\sigma) = l(\tilde{\sigma})$. To prove this, let $T_1 = \log\left(\frac{p}{q}\right)$ and $T_2 = \log\left(\frac{1-p}{1-q}\right)$; then,

$$\begin{aligned} l(\sigma) &= \left(T_1 \sum_{i_1 i_2 \ldots i_m} \mathbb{1}[A_{i_1 i_2 \ldots i_m} = 1] + T_2 \sum_{i_1 i_2 \ldots i_m} \mathbb{1}[A_{i_1 i_2 \ldots i_m} = 0]\right) \mathbb{1}[\sigma_{i_1} = \cdots = \sigma_{i_m} = +1] \\ &+ \left(T_1 \sum_{i_1 i_2 \ldots i_m} \mathbb{1}[A_{i_1 i_2 \ldots i_m} = 1] + T_2 \sum_{i_1 i_2 \ldots i_m} \mathbb{1}[A_{i_1 i_2 \ldots i_m} = 0]\right) \mathbb{1}[\sigma_{i_1} = \cdots = \sigma_{i_m} = -1]. \end{aligned}$$

Further, $l(\sigma)$ can be written as

$$\begin{aligned} l(\sigma) &= T_1 \sum_{\substack{i_1 i_2 \ldots i_m \in I_+(\sigma) \\ i_1 i_2 \ldots i_m \neq i_0}} \mathbb{1}[A_{i_1 i_2 \ldots i_m} = 1] + T_1 \sum_{\substack{i_2 \ldots i_m \in I_+(\sigma) \\ i_2 \ldots i_m \neq i_0}} \mathbb{1}[A_{i_0 i_2 \ldots i_m} = 1] \\ &+ T_2 \sum_{\substack{i_1 i_2 \ldots i_m \in I_+(\sigma) \\ i_1 i_2 \ldots i_m \neq i_0}} \mathbb{1}[A_{i_1 i_2 \ldots i_m} = 0] + T_2 \sum_{\substack{i_2 \ldots i_m \in I_+(\sigma) \\ i_2 \ldots i_m \neq i_0}} \mathbb{1}[A_{i_0 i_2 \ldots i_m} = 0] \\ &+ T_1 \sum_{\substack{i_1 i_2 \ldots i_m \in I_-(\sigma) \\ i_1 i_2 \ldots i_m \neq j_0}} \mathbb{1}[A_{i_1 i_2 \ldots i_m} = 1] + T_1 \sum_{\substack{i_2 \ldots i_m \in I_-(\sigma) \\ i_2 \ldots i_m \neq j_0}} \mathbb{1}[A_{j_0 i_2 \ldots i_m} = 1] \\ &+ T_2 \sum_{\substack{i_1 i_2 \ldots i_m \in I_-(\sigma) \\ i_1 i_2 \ldots i_m \neq j_0}} \mathbb{1}[A_{i_1 i_2 \ldots i_m} = 0] + T_2 \sum_{\substack{i_2 \ldots i_m \in I_-(\sigma) \\ i_2 \ldots i_m \neq j_0}} \mathbb{1}[A_{j_0 i_2 \ldots i_m} = 0], \end{aligned}$$

and

$$\begin{aligned} l(\tilde{\sigma}) &= T_1 \sum_{\substack{i_1 i_2 \ldots i_m \in I_+(\sigma) \\ i_1 i_2 \ldots i_m \neq j_0}} \mathbb{1}[A_{i_1 i_2 \ldots i_m} = 1] + T_1 \sum_{\substack{i_2 \ldots i_m \in I_+(\sigma) \\ i_2 \ldots i_m \neq j_0}} \mathbb{1}[A_{j_0 i_2 \ldots i_m} = 1] \\ &+ T_2 \sum_{\substack{i_1 i_2 \ldots i_m \in I_+(\sigma) \\ i_1 i_2 \ldots i_m \neq j_0}} \mathbb{1}[A_{i_1 i_2 \ldots i_m} = 0] + T_2 \sum_{\substack{i_2 \ldots i_m \in I_+(\sigma) \\ i_2 \ldots i_m \neq j_0}} \mathbb{1}[A_{j_0 i_2 \ldots i_m} = 0] \\ &+ T_1 \sum_{\substack{i_1 i_2 \ldots i_m \in I_-(\sigma) \\ i_1 i_2 \ldots i_m \neq i_0}} \mathbb{1}[A_{i_1 i_2 \ldots i_m} = 1] + T_1 \sum_{\substack{i_2 \ldots i_m \in I_-(\sigma) \\ i_2 \ldots i_m \neq i_0}} \mathbb{1}[A_{i_0 i_2 \ldots i_m} = 1] \\ &+ T_2 \sum_{\substack{i_1 i_2 \ldots i_m \in I_-(\sigma) \\ i_1 i_2 \ldots i_m \neq i_0}} \mathbb{1}[A_{i_1 i_2 \ldots i_m} = 0] + T_2 \sum_{\substack{i_2 \ldots i_m \in I_-(\sigma) \\ i_2 \ldots i_m \neq i_0}} \mathbb{1}[A_{i_0 i_2 \ldots i_m} = 0] \end{aligned}$$

Then $l(\sigma) = l(\tilde{\sigma})$ by the assumption of i_0 and j_0.

Next, we show there are k_n ($k_n \to \infty$) such pairs. More specifically, we show that there exist $i_1, i_2, \ldots, i_k \in I_+(\sigma)$ and $j_1, j_2, \ldots, j_k \in I_-(\sigma)$ with $k \gg 1$ such that the likelihood function remains unchanged if we flip the label of a pair (i_t, j_t), for $t = 1, 2, \ldots, k$. Let η_t be the label obtained by flipping the label of i_t, j_t in σ. Then, $l(\eta_t) = l(\sigma)$, for $1 \le t \le k \to \infty$.

Let $n_1 = |I_+(\sigma)|$ and $n_2 = |I_-(\sigma)|$. Then, $n_1, n_2 = \frac{n}{2}(1 + O(n^{-\frac{1}{3}}))$ with probability $1 - o(1)$. Hence, we take $n_1 = n_2 = \frac{n}{2}$ below. Let $S_+ \subset I_+(\sigma)$ be a random subset with $|S_+| = \frac{n}{\log^2 n}$, and $S_- \subset I_-(\sigma)$ be a random subset with $|S_-| = \frac{n}{\log^2 n}$. Denote $S = S_+ \cup S_-$. Define

$$S_0 = \left\{ i \in S | any\ i_2, \ldots, i_t \in S, i_{t+1}, \ldots, i_m \in S^c,\ s.t.\ A_{ii_2 \ldots i_t i_{t+1} \ldots i_m} = *,\ t \ge 2 \right\}.$$

For each node $i \in S_0$, the hyperedge $A_{ii_2 \ldots i_m}$ is possibly revealed if and only if $\{i_2, \ldots, i_m\} \subset I_+(\sigma) - S$ or $\{i_2, \ldots, i_m\} \subset I_-(\sigma) - S$.

We will show $|S_0| = \frac{2n(1+o(1))}{\log^2 n}$ with probability $1 - o(1)$. Let

$$T = \sum_{t=2}^{m} \sum_{\substack{i_1, \ldots, i_t \in S \\ i_{t+1}, \ldots, i_m \in S^c}} \mathbb{1}[A_{i_1 i_2 \ldots i_t i_{t+1} \ldots i_m} \neq *].$$

The expectation of T is

$$\begin{aligned}
\mathbb{E}T &= \sum_{t=2}^{m} \binom{\frac{2n}{\log^2 n}}{t} \binom{n - \frac{2n}{\log^2 n}}{m - t} \alpha \\
&= \sum_{t=2}^{m} \binom{\frac{2n}{\log^2 n}}{t} \binom{n - \frac{2n}{\log^2 n}}{m - t} \frac{t \log n}{n^{m-1}} \\
&= \frac{c \cdot n^m}{\log^4 n} \frac{t \log n}{n^{m-1}} \\
&\asymp \frac{n}{\log^3 n}.
\end{aligned}$$

Hence, by the Markov inequality, we have

$$\mathbb{P}\left(T \ge \frac{n}{\log^2 n \sqrt{\log n}} \right) \le \frac{1}{\frac{n}{\log^2 n \sqrt{\log n}}} \frac{c \cdot n}{\log^3 n} = \frac{\sqrt{\log n}}{\log n} = o(1).$$

Then, $T < \frac{n}{\log^2 n \sqrt{\log n}}$ with probablity $1 - o(1)$. Hence, $|S_0| = \frac{2n}{\log^2 n}(1 + o(1))$ with probability $1 - o(1)$.

Let $m_1 = \frac{\sqrt{pq}t\log n}{2^{m-1}(m-1)!}$ and $m_2 = \frac{\sqrt{(1-p)(1-q)}t\log n}{2^{m-1}(m-1)!}$. For some $k \gg 1$, we show that there exists $i_t \in S_0 \cap S_+,\ (1 \le t \le k)$ such that i_t has m_1 present hyperedges and m_2 absent hyperedges in $I_+(\sigma)$ and $I_-(\sigma)$ respectively. Denote

$$\widetilde{n}_1 = \tbinom{n_1 - \frac{2n}{\log^2 n}}{m-1} \sim \frac{n^{m-1}}{2^{m-1}(m-1)!}.$$

Let $i_0 \in S_0 \cap S_+$, the probability that i_0 has m_1 present hyperedges and m_2 absent hyperedges in $I_+(\sigma)$ and $I_-(\sigma)$ is,

$$\begin{aligned}
p_0 &= \frac{\widetilde{n}_1!}{m_1!\,m_2!\,(\widetilde{n}_1 - m_1 - m_2)!} \cdot (\alpha p)^{m_1}[\alpha(1-p)]^{m_2}(1-\alpha)^{(\widetilde{n}_1 - m_1 - m_2)} \\
&\quad \times \frac{\widetilde{n}_1!}{m_1!\,m_2!\,(\widetilde{n}_1 - m_1 - m_2)!} \cdot (\alpha q)^{m_1}[\alpha(1-q)]^{m_2}(1-\alpha)^{(\widetilde{n}_1 - m_1 - m_2)} \\
&\sim \frac{1}{m_1!^2\,m_2!^2}\left[\frac{\widetilde{n}_1^{\widetilde{n}_1+\frac{1}{2}}e^{-\widetilde{n}_1}}{(\widetilde{n}_1 - m_1 - m_2)^{\widetilde{n}_1 - m_1 - m_2 + \frac{1}{2}}e^{-\widetilde{n}_1 + m_1 + m_2}}\right]^2 (\alpha^2 pq)^{m_1} \\
&\times [\alpha^2(1-p)(1-q)]^{m_2}(1-\alpha)^{2(\widetilde{n}_1 - m_1 - m_2)} \\
&= \frac{1}{m_1!^2\,m_2!^2}\left[\frac{(\widetilde{n}_1 - m_1 - m_2)^{m_1+m_2}}{e^{m_1+m_2}(1 - \frac{m_1+m_2}{\widetilde{n}_1})^{\widetilde{n}_1+\frac{1}{2}}}\right]^2 (\alpha^2 pq)^{m_1}[\alpha^2(1-p)(1-q)]^{m_2}(1-\alpha)^{2(\widetilde{n}_1 - m_1 - m_2)} \\
&= \frac{1}{m_1!^2\,m_2!^2}\left[\frac{\widetilde{n}_1^{m_1+m_2}}{e^{m_1+m_2}e^{-(m_1+m_2)}}\right]^2 (\alpha^2 pq)^{m_1}\left[\alpha^2(1-p)(1-q)\right]^{m_2} e^{-\frac{t\log n}{2^{m-2}(m-1)!}} \\
&= \frac{\tilde{n}_1^{2(m_1+m_2)}}{m_1!^2\,m_2!^2} e^{-\frac{t\log n}{2^{m-2}(m-1)!}} (\alpha^2 pq)^{m_1}[\alpha^2(1-p)(1-q)]^{m_2} \\
&= \frac{n^{-\frac{t}{2^{m-2}(m-1)!}}}{m_1!^2\,m_2!^2}(\alpha^2\tilde{n}_1^2 pq)^{m_1}[\alpha^2\tilde{n}_1^2(1-p)(1-q)]^{m_2} \\
&= n^{-\frac{t}{2^{m-2}(m-1)!}}\frac{e^{2(m_1+m_2)}}{4\pi^2 m_1 m_2}(\frac{\alpha^2\tilde{n}_1^2 pq}{m_1^2})^{m_1}(\frac{\alpha^2\tilde{n}_1^2(1-p)(1-q)}{m_2^2})^{m_2} \\
&= \frac{1}{4\pi^2 m_1 m_2} n^{-\frac{t}{2^{m-2}(m-1)!}} e^{\frac{\sqrt{pq}+\sqrt{(1-p)(1-q)}}{2^{m-2}(m-1)!}t\log n} \\
&= \frac{1}{4\pi^2 m_1 m_2} n^{-\frac{t}{2^{m-2}(m-1)!}[1-\sqrt{pq}-\sqrt{(1-p)(1-q)}]} \\
&= \frac{1}{4\pi^2 m_1 m_2} n^{-t\cdot\frac{(\sqrt{p}-\sqrt{q})^2+(\sqrt{1-p}-\sqrt{1-q})^2}{2^{m-1}(m-1)!}}.
\end{aligned}$$

If $t < \frac{2^{m-1}(m-1)!}{(\sqrt{p}-\sqrt{q})^2+(\sqrt{1-p}-\sqrt{1-q})^2}$, then $p_0 \gg \frac{n^{1-\epsilon}}{n}$, for some $\epsilon \in (0,1)$. Similarly, the probability that $j_0 \in S_0 \cap S_-$ has m_1 present hyperedges and m_2 absent hyperedges in $I_+(\sigma)$ and $I_-(\sigma)$ is equal to p_0.

For $i \in S_0$, let $\mathbb{1}_i$ denote the event that i has m_1 present hyperedges and m_2 absent hyperedges in $I_+(\sigma)$ and $I_-(\sigma)$. Define two random variables

$$X = \sum_{i \in S_0 \cap S_+} \mathbb{1}_i, \qquad Y = \sum_{i \in S_0 \cap S_-} \mathbb{1}_i.$$

If $\mathbb{1}_i = \mathbb{1}_j = 1$ for $i \in S_0 \cap S_+$ and $j \in S_0 \cap S_-$, then the likelihood function remains unchanged if we flip the labels of i and j. By Chebyshev's inequality, given $|S_0 \cap S_+|$, we have

$$\begin{aligned}
& \mathbb{P}\left(X \le (1-\epsilon)\frac{2n}{\log^2 n} p_0\right) \\
= \; & \mathbb{P}\left(X \le (1-\epsilon)\frac{2n}{\log^2 n} p_0 \middle| |S_0 \cap S_+| \ge \frac{2n}{\log^2 n}(1-o(1))\right) \cdot \mathbb{P}\left(|S_0 \cap S_+| \ge \frac{2n}{\log^2 n}(1-o(1))\right) \\
+ \; & \mathbb{P}\left(X \le (1-\epsilon)\frac{2n}{\log^2 n} p_0 \middle| |S_0 \cap S_+| < \frac{2n}{\log^2 n}(1-o(1))\right) \mathbb{P}\left(|S_0 \cap S_+| < \frac{2n}{\log^2 n}\right) \\
\le \; & \mathbb{P}\left(X \le (1-\epsilon)|S_0 \cap S_+| p_0 \middle| |S_0 \cap S_+| \ge \frac{2n}{\log^2 n}(1-o(1))\right) + o(1) \\
\le \; & \frac{1}{\epsilon^2 |S_0 \cap S_+| p_0} + o(1).
\end{aligned}$$

Given that $p_0 \gg \frac{n^{1-\epsilon}}{n}$ for some $\epsilon > 0$ and $|S_0 \cap S_+| \ge \frac{2n}{\log^2 n}(1-o(1))$, we observe that $X \ge |S_0 \cap S_+| p_0 \to +\infty$ with probability $1-o(1)$. Similarly, $Y \ge |S_0 \cap S_+| p_0 \to +\infty$ with probability $1-o(1)$. Consequently, we can establish the existence of pairs (i_t, j_t) for $1 \le t \le k_n \to \infty$. For each t, flipping the labels of i_t and j_t maintains the likelihood constant. By invoking Proposition 4.1 in Dhara et al. (2021), the proof is completed.

3.2. Proof of Theorem 2.2

Proof of Theorem 2.2 : Let σ denote the MLE. Referring to the log-likelihood function mentioned in (3.1), we ascertain that the MLE fails to achieve exact recovery if there exists a label η for which $l(\eta) \ge l(\sigma)$ with a probability of at least δ, where δ is a positive constant. To establish the desired result, our proof focuses on demonstrating that the probability of MLE failure is $o(1)$.

We derive the MLE by maximizing $\log L$ in (3.1) with respect to σ. It is worth noting that the first term in $\log L$ does not depend on σ. Therefore, to obtain the MLE, we only need to maximize the second term in $\log L$. Let σ represent the MLE. Recall that the MLE fails if there exists a label η for which $l(\eta) \ge l(\sigma)$ with a probability of at least δ, where δ is a positive constant. In the following, we demonstrate that the probability of MLE failure is $o(1)$.

Let k be an even number and $1 \leq k \leq \frac{n}{2}$. Define the Hamming distance between two labels σ and η as

$$d(\sigma, \eta) = \min \left\{ \sum_{i=1}^{n} \mathbb{1}[\sigma_i \neq \eta_i], \sum_{i=1}^{n} \mathbb{1}[\sigma_i \neq -\eta_i] \right\}.$$

Let η be a label such that $d(\sigma, \eta) = k$, and denote

$$C_{i_1 i_2 \ldots i_m}(A) = \log\left(\frac{p}{q}\right) \mathbb{1}[A_{i_1 i_2 \ldots i_m} = 1] + \log\left(\frac{1-p}{1-q}\right) \mathbb{1}[A_{i_1 i_2 \ldots i_m} = 0].$$

Then, log-likelihood difference at η and σ is

$$l(\eta) - l(\sigma) = \sum_{1 \leq i_1 < \cdots < i_m \leq n} C_{i_1 i_2 \ldots i_m}(A)(\mathbb{1}_{i_1 \ldots i_m}(\eta) - \mathbb{1}_{i_1 \ldots i_m}(\sigma)).$$

We show that

$$\mathbb{P}(\exists k \; and \; d(\sigma, \eta) = k \; , s.t. \; l(\eta) - l(\sigma) \geq 0) = o(1).$$

Recall $\mathrm{I}_+(\sigma)$ and $\mathrm{I}_-(\sigma)$. Denote $\mathbb{1}_{i_1 \ldots i_m}(\eta) = I[\eta_{i_1} = \eta_{i_2} = \cdots = \eta_{i_m}]$. Note that

$$\mathbb{1}_{i_1 \ldots i_m}(\eta) - \mathbb{1}_{i_1 \ldots i_m}(\sigma) = \begin{cases} 1, & i_1 \ldots i_m \subset \mathrm{I}_+(\eta) \; or \; \mathrm{I}_-(\eta), \; i_1 \ldots i_m \not\subset \mathrm{I}_+(\sigma) \; , \mathrm{I}_-(\sigma); \\ -1, & i_1 \ldots i_m \subset \mathrm{I}_+(\sigma) \; or \; \mathrm{I}_-(\sigma), \; i_1 \ldots i_m \not\subset \mathrm{I}_+(\eta), \mathrm{I}_-(\eta); \\ 0, & otherwise. \end{cases}$$

Hence, $l(\eta) - l(\sigma)$ is written as

$$l(\eta) - l(\sigma) = \sum_{\substack{i_1 \ldots i_m \\ i_1 \ldots i_m \subset I_+(\eta) \; or \; \mathrm{I}_-(\eta) \\ i_1 \ldots i_m \not\subset I_+(\sigma), I_-(\sigma)}} C_{i_1 \ldots i_m}(A) - \sum_{\substack{i_1 \ldots i_m \\ i_1 \ldots i_m \subset I_+(\sigma) \; or \; \mathrm{I}_-(\sigma) \\ i_1 \ldots i_m \not\subset I_+(\eta), I_-(\eta)}} C_{i_1 \ldots i_m}(A).$$

It is easy to verify that there are $n_k = 2\left[\binom{\frac{n}{2}}{m} - \binom{\frac{k}{2}}{m} - \binom{\frac{n-k}{2}}{m}\right]$ hyperedges $\{i_1, \ldots, i_m\}$ such that $\{i_1 \ldots i_m\} \subset \mathbb{1}_+(\eta)$ or $\mathbb{1}_-(\eta)$ and $\{i_1 \ldots i_m\} \not\subset \mathbb{1}_+(\sigma), \mathbb{1}_-(\sigma)$. For convenience, define random variables X and Y as

$$\mathbb{P}(X = 1) = \alpha p, \quad \mathbb{P}(X = 0) = \alpha(1-p), \quad \mathbb{P}(X = -1) = 1 - \alpha.$$

$$\mathbb{P}(Y = 1) = \alpha q, \quad \mathbb{P}(Y = 0) = \alpha(1-q), \quad \mathbb{P}(Y = -1) = 1 - \alpha.$$

Let X_i, Y_i be independent and identically distributed (i.i.d) copies of X, Y, respectively, and

$$W_i = \log\left(\frac{p}{q}\right)\mathbb{1}[X_i = 1] + \log\left(\frac{1-p}{1-q}\right)\mathbb{1}[X_i = 0]$$
$$V_i = \log\left(\frac{p}{q}\right)\mathbb{1}[Y_i = 1] + \log\left(\frac{1-p}{1-q}\right)\mathbb{1}[Y_i = 0].$$

For any $r > 0$, by the Markov inequality, we have

$$\begin{aligned}
\mathbb{P}(l(\eta) - l(\sigma) \geq 0) &= \mathbb{P}\left(\sum_{i=1}^{n_k}(V_i - W_i) \geq 0\right) \\
&= \mathbb{P}\left(\sum_{i=1}^{n_k}(W_i - V_i) \leq 0\right) \\
&= \mathbb{P}\left(e^{\sum_{i=1}^{n_k}(-r)(W_i - V_i)} \geq 1\right) \\
&\leq \left[\mathbb{E}\left(e^{-rW_1}\right)\mathbb{E}\left(e^{rV_1}\right)\right]^{n_k}.
\end{aligned}$$

Next, we find the explicit expression of expectations $\mathbb{E}\left(e^{-rW_1}\right)$ and $\mathbb{E}\left(e^{rV_1}\right)$.

$$\begin{aligned}
\mathbb{E}[e^{-rW_1}] &= \mathbb{E}e^{-r\left(\log\left(\frac{p}{q}\right)\mathbb{1}[X_i=1] + \log\left(\frac{1-p}{1-q}\right)\mathbb{1}[X_i=0]\right)} \\
&= e^{-r\log(\frac{p}{q})}\alpha p + e^{-r\log(\frac{1-p}{1-q})}\alpha(1-p) + (1-\alpha) \\
&= \left(\frac{q}{p}\right)^r \alpha p + \left(\frac{1-q}{1-p}\right)^r \alpha(1-p) + (1-\alpha)
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}[e^{rV_1}] &= \mathbb{E}e^{r\left(\log(\frac{p}{q})\mathbb{1}[Y_i=1] + \log(\frac{1-p}{1-q})\mathbb{1}[Y_i=0]\right)} \\
&= e^{r\log(\frac{p}{q})}\alpha q + e^{r\log(\frac{1-p}{1-q})}\alpha(1-q) + (1-\alpha) \\
&= \left(\frac{p}{q}\right)^r \alpha q + \left(\frac{1-p}{1-q}\right)^r \alpha(1-q) + (1-\alpha)
\end{aligned}$$

Taking $r = \frac{1}{2}$ yields

$$\begin{aligned}
\mathbb{E}[e^{-rW_1}] &= \alpha\sqrt{pq} + \alpha\sqrt{(1-p)(1-q)} + (1-\alpha) \\
&= 1 + \alpha[\sqrt{pq} + \sqrt{(1-p)(1-q)} - 1], \\
\mathbb{E}[e^{rV_1}] &= \alpha\sqrt{pq} + \alpha\sqrt{(1-p)(1-q)} + (1-\alpha) \\
&= 1 + \alpha[\sqrt{pq} + \sqrt{(1-p)(1-q)} - 1].
\end{aligned}$$

Hence,

$$\begin{aligned}
\log \mathbb{P}\left(l(\eta) - l(\sigma) \geq 0\right) &\leq n_k \log \mathbb{E}[e^{-rW_1}] + n_k \log \mathbb{E}[e^{rV_1}] \\
&\leq n_k[2\alpha(\sqrt{pq} + \sqrt{(1-p)(1-q)} - 1)] \\
&= n_k\alpha\left[(-1)\left\{(\sqrt{p}-\sqrt{q})^2 + (\sqrt{1-p}-\sqrt{1-q})^2\right\}\right] \\
&= -n_k\alpha\left[(\sqrt{p}-\sqrt{q})^2 + (\sqrt{1-p}-\sqrt{1-q})^2\right]. \qquad (3.2)
\end{aligned}$$

For $k \geq \frac{n}{\log\log n}$, it is easy to check that $n_k \geq \frac{1}{2^{m-1}} \frac{n}{\log\log n}\binom{n-1}{m-1}$. Hence by (3.2), we obtain

$$\begin{aligned}
\mathbb{P}\left(l(\eta) - l(\sigma) \geq 0\right) &\leq e^{-[(\sqrt{p}-\sqrt{q})^2 + (\sqrt{1-p}-\sqrt{1-q})^2]\frac{t\log n}{n^{m-1}}\frac{1}{2^{m-1}}\frac{n}{\log\log n}\frac{n^{m-1}}{(m-1)!}} \\
&= e^{-[(\sqrt{p}-\sqrt{q})^2 + (\sqrt{1-p}-\sqrt{1-q})^2]\frac{t}{2^{m-1}(m-1)!}\frac{n\log n}{\log\log n}} \\
&= e^{-c\frac{n\log n}{\log\log n}},
\end{aligned}$$

for some positive constant c. Clearly, there are $\binom{\frac{n}{2}}{\frac{k}{2}}^2$ choices for η, with $d(\sigma,\eta) = k$. Note that $\binom{\frac{n}{2}}{\frac{k}{2}}^2 \leq 2^n$. Then, the probability that there exists η with $d(\sigma,\eta) = k$ for $k \geq \frac{n}{\log\log n}$ is upper bounded by

$$\frac{n}{2} \cdot 2^n \cdot e^{-c\frac{n\log n}{\log\log n}} = e^{n\log 2 + \log\frac{n}{2} - cn\frac{\log n}{\log\log n}} = o(1).$$

For $k < \frac{n}{\log\log n}$, we have $n_k = \frac{k}{2^{m-1}}\binom{n-1}{m-1}$. Then

$$\begin{aligned}
\mathbb{P}\left(l(\eta) - l(\sigma) \geq 0\right) &\leq e^{-[(\sqrt{p}-\sqrt{q})^2 + (\sqrt{1-p}-\sqrt{1-q})^2]\frac{t\log n}{n^{m-1}}\frac{k}{2^{m-1}}\frac{n^{m-1}}{(m-1)!}} \\
&= e^{-\frac{(\sqrt{p}-\sqrt{q})^2 + (\sqrt{1-p}-\sqrt{1-q})^2}{2^{m-1}(m-1)!}tk\log n} \\
&= n^{-\frac{[\sqrt{p}-\sqrt{q}]^2 + [\sqrt{1-p}-\sqrt{1-q}]^2}{2^{m-1}(m-1)!}tk}.
\end{aligned}$$

There are $\binom{\frac{n}{2}}{\frac{k}{2}}^2 \leq n^k$ choices for η, with $d(\sigma,\eta) = k$. Then the probability that there exists η with $d(\sigma,\eta) = k$ for $k < \frac{n}{\log\log n}$ is upper bounded by

$$\begin{aligned}
k \cdot \binom{\frac{n}{2}}{\frac{k}{2}}^2 \mathbb{P}\left(l(\eta) - l(\sigma) \geq 0\right) &\leq kn^k \cdot n^{-\frac{[\sqrt{p}-\sqrt{q}]^2 + [\sqrt{1-p}-\sqrt{1-q}]^2}{2^{m-1}(m-1)!}tk} \\
&\leq kn^k n^{-(1+\epsilon)k} \\
&= \frac{k}{n^{\epsilon k}} = o(1),
\end{aligned}$$

where ϵ is a constant such that $\frac{[\sqrt{p}-\sqrt{q}]^2+[\sqrt{1-p}-\sqrt{1-q}]^2}{2^{m-1}(m-1)!}t = 1+\epsilon$. This is possible by the condition $t > \frac{2^{m-1}(m-1)!}{(\sqrt{p}-\sqrt{q})^2+(\sqrt{1-p}-\sqrt{1-q})^2}$. The proof is complete.

3.3. Proof of Theorem 2.3

The proof proceeds by showing the probability that there exists a mislabeled node goes to zero. By the definition of the hypergraph $\tilde{A}$, we have

$$\begin{aligned}
\mathbb{P}(\tilde{A}_{i_1 i_2 \ldots i_m} = 1) &= \begin{cases} \frac{\log\log n}{\log n} \cdot \alpha p, & \{i_1, i_2, \ldots, i_m\} \subset I_+(\sigma) \text{ or } I_-(\sigma), \\ \frac{\log\log n}{\log n} \cdot \alpha q, & \text{otherwise.} \end{cases} \\
&= \begin{cases} \frac{tp\log\log n}{n^{m-1}}, & \{i_1, i_2, \ldots, i_m\} \subset I_+(\sigma) \text{ or } I_-(\sigma), \\ \frac{tq\log\log n}{n^{m-1}}, & \text{otherwise.} \end{cases}
\end{aligned}$$

As a result of the transformation, the hypergraph $\tilde{A}$ maintains the same community structure as the original hypergraph A. In $\tilde{A}$, the order of hyperedge probabilities is $\frac{\log\log n}{n^{m-1}}$. With a probability of $1-o(1)$, the weak recovery algorithm described in Ahn et al. (2018) successfully recovers the true labels of $(1-\delta)n$ nodes in $\tilde{A}$, where $\delta = o(1)$. Denoting the resulting communities as $\tilde{I}+(\sigma)$ and $\tilde{I}-(\sigma)$, it follows that, with a probability of $1-o(1)$, $\frac{\delta}{2}n$ nodes in $\tilde{I}+(\sigma)$ and $\tilde{I}-(\sigma)$ are mislabeled. In the refinement step, a node i among the correctly labeled $\frac{1-\delta}{2}n$ nodes in $\tilde{I}_+(\sigma)$ is mislabeled if

$$e(i, \tilde{I}_+(\sigma)) < e(i, \tilde{I}_-(\sigma)).$$

A node among the mislabeled $\frac{\delta}{2}n$ nodes in $\tilde{I}_+(\sigma)$ remains mislabeled if

$$e(i, \tilde{I}_+(\sigma)) \geq e(i, \tilde{I}_-(\sigma)).$$

A similar result holds for nodes in $\tilde{I}_-(\sigma)$. Let X_i, Y_i, W_i and V_i be defined as in the proof of Theorem 2.2, and let W_i' and V_i' be *i.i.d.* copies of W_i and V_i, respectively. Then, a node i being mislabeled is equivalent to

$$\sum_{i=1}^{\binom{\frac{\delta}{2}n}{m-1}} W_i + \sum_{i=1}^{\binom{\frac{n}{2}}{m-1}-\binom{\frac{\delta}{2}n}{m-1}} V_i \geq \sum_{i=1}^{\binom{(1-\delta)\frac{n}{2}}{m-1}} W_i' + \sum_{i=1}^{\binom{\frac{n}{2}}{m-1}-\binom{(1-\delta)\frac{n}{2}}{m-1}} V_i'.$$

We bound the probability that node i is mislabeled and then apply the union bound. Let $r = \frac{1}{\delta\sqrt{\log(\frac{1}{\delta})}}$. Then, we have

$$
\begin{aligned}
p_i &= \mathbb{P}(\text{node } i \text{ is mislabelled}) \\
&= \mathbb{P}\left[\sum_{i=1}^{\binom{\frac{\delta}{2}n}{m-1}} W_i + \sum_{i=1}^{\binom{\frac{n}{2}}{m-1}-\binom{\frac{\delta}{2}n}{m-1}} V_i \geq \sum_{i=1}^{\binom{(1-\delta)\frac{n}{2}}{m-1}} W_i' + \sum_{i=1}^{\binom{\frac{n}{2}}{m-1}-\binom{(1-\delta)\frac{n}{2}}{m-1}} V_i'\right] \\
&= \mathbb{P}\left[\sum_{i=1}^{\binom{\frac{n}{2}}{m-1}-\binom{\frac{\delta}{2}n}{m-1}} (V_i - W_i') + \sum_{i=1}^{\binom{\frac{\delta}{2}n}{m-1}} W_i \geq \sum_{i=1}^{\binom{\frac{n}{2}}{m-1}-\binom{(1-\delta)\frac{n}{2}}{m-1}} V_i' - \sum_{i=1}^{\binom{\frac{n}{2}}{m-1}-\binom{\frac{\delta n}{2}}{m-1}-\binom{(1-\delta)\frac{n}{2}}{m-1}} W_i'\right] \\
&\leq \mathbb{P}\left[\sum_{i=1}^{\binom{\frac{n}{2}}{m-1}-\binom{\frac{\delta}{2}n}{m-1}} (V_i - W_i') \geq -r\delta\log n\right] + \\
&\quad \mathbb{P}\left[\sum_{i=1}^{\binom{\frac{\delta}{2}n}{m-1}} W_i + \sum_{i=1}^{\binom{\frac{n}{2}}{m-1}-\binom{\frac{\delta n}{2}}{m-1}-\binom{(1-\delta)\frac{n}{2}}{m-1}} W_i' - \sum_{i=1}^{\binom{\frac{n}{2}}{m-1}-\binom{(1-\delta)\frac{n}{2}}{m-1}} V_i' \geq r\delta\log n\right] \\
&= (I) + (II).
\end{aligned}
$$

Next, we show $(II) = O\left(n^{-2}\right)$ and $(I) = O\left(n^{-\frac{t}{I_m(p,q)}}\right)$. It is easy to verify that

$$
\begin{aligned}
(II) &\leq \mathbb{P}\left(\sum_{i=1}^{\binom{\frac{\delta}{2}n}{m-1}} W_i \geq \frac{r\delta}{3}\log n\right) + \mathbb{P}\left(\sum_{i=1}^{\binom{\frac{n}{2}}{m-1}-\binom{\frac{\delta n}{2}}{m-1}-\binom{(1-\delta)\frac{n}{2}}{m-1}} W_i' \geq \frac{r\delta}{3}\log n\right) \\
&\quad + \mathbb{P}\left(\sum_{i=1}^{\binom{\frac{n}{2}}{m-1}-\binom{(1-\delta)\frac{n}{2}}{m-1}} -V_i' \geq \frac{r\delta}{3}\log n\right).
\end{aligned}
$$

Because $p > q > 0$, it follows that $1 - q > 1 - p$, and then

$$
\begin{aligned}
W_i &= \log\left(\frac{p}{q}\right)\mathbb{1}[X_i = 1] + \log\left(\frac{1-p}{1-q}\right)\mathbb{1}[X_i = 0] \\
&\leq \log\left(\frac{p}{q}\right)\mathbb{1}[X_i = 1].
\end{aligned}
$$

Then, by the multiplicative Chernoff bound, we have

$$\begin{aligned}\mathbb{P}\left(\sum_{i=1}^{\binom{\frac{\delta}{2}n}{m-1}} W_i \geq \frac{r\delta}{3}\log n\right) &\leq \mathbb{P}\left(\sum_{i=1}^{\binom{\frac{\delta}{2}n}{m-1}} \mathbb{1}[X_i=1] \geq \frac{r\delta\log n}{3\log(\frac{p}{q})}\right)\\ &\leq \left(\frac{\frac{r}{\delta^{m-2}}2^{m-1}(m-1)!}{e\cdot 3pt\log(\frac{p}{q})}\right)^{-\frac{r\delta\log n}{3\log(\frac{p}{q})}}\end{aligned}$$

$$\begin{aligned}&= e^{-\frac{\log n}{3\log(\frac{p}{q})\sqrt{\log(\frac{1}{\delta})}}[\log(\frac{1}{\delta})+(m-2)\log(\frac{1}{\delta})(1+o(1))]}\\ &= e^{-\frac{(m-1)\sqrt{\log(\frac{1}{\delta})}}{3\log(\frac{p}{q})}\log n\,(1+o(1))}\\ &= O\left(n^{-2}\right).\end{aligned}$$

Similarly, we have

$$\mathbb{P}\left(\sum_{i=1}^{\binom{\frac{n}{2}}{m-1}-\binom{\frac{\delta n}{2}}{m-1}-\binom{(1-\delta)\frac{n}{2}}{m-1}} W_i' \geq \frac{r\delta}{3}\log n\right) = O\left(n^{-2}\right).$$

Note that

$$\begin{aligned}-V_i' &= \log\left(\frac{1-p}{1-q}\right)\mathbb{1}[A_i=0] - \log\left(\frac{p}{q}\right)\mathbb{1}[A_i=1]\\ &\leq \log\left(\frac{1-p}{1-q}\right)\mathbb{1}[A_i=0].\end{aligned}$$

Hence, by the multiplicative Chernoff bound, it follows that

$$\begin{aligned}\mathbb{P}\left(\sum_{i=1}^{\binom{\frac{n}{2}}{m-1}-\binom{(1-\delta)\frac{n}{2}}{m-1}} (-V_i') \geq \frac{r\delta}{3}\log n\right) &\leq \mathbb{P}\left(\sum_{i=1}^{\binom{\frac{n}{2}}{m-1}-\binom{(1-\delta)\frac{n}{2}}{m-1}} \mathbb{1}[A_i=0] \geq \frac{r\delta\log n}{3\log(\frac{1-q}{1-p})}\right)\\ &\leq \left(\frac{\frac{r}{\delta^{m-2}}2^{m-1}(m-1)!}{e\cdot 3(1-p)t\log(\frac{1-q}{1-p})}\right)^{-\frac{r\delta\log n}{3\log(\frac{1-q}{1-p})}}\\ &= e^{-\frac{1-\delta\log n}{3\log(\frac{1-q}{1-p})}[(m-1)\log(\frac{1}{\delta})(1+o(1))]}\\ &= e^{-\frac{(m-1)\sqrt{\log(\frac{1}{\delta})}\log n}{3\log(\frac{1-q}{1-p})}(1+o(1))}\\ &= O\left(n^{-2}\right).\end{aligned}$$

Thus, we conclude that $(II) = O\left(n^{-2}\right)$.

Next, we bound (I). Note that $\binom{\frac{n}{2}}{m-1} - \binom{\frac{\delta}{2}n}{m-1} = \frac{n^{m-1}}{2^{m-1}(m-1)!}(1+o(1))$. By Markov's inequality, we have

$$\begin{aligned}
(I) &= \mathbb{P}\left[e^{\sum_{i=1}^{\binom{\frac{n}{2}}{m-1}-\binom{\frac{\delta}{2}n}{m-1}}(V_i - W_i')} \geq e^{-\frac{r\delta\log n}{2}}\right] \\
&\leq e^{r\delta\frac{\log n}{2}}\left(\mathbb{E}[e^{\frac{1}{2}V_1}e^{-\frac{1}{2}W_1'}]\right)^{\frac{n^{m-1}}{2^{m-1}(m-1)!}} \\
&= e^{r\delta\frac{\log n}{2}}[e^{-\frac{1}{2}\log(\frac{p}{q})}\alpha p + e^{-\frac{1}{2}\log(\frac{1-p}{1-q})}\alpha(1-p) + (1-\alpha)]^{\frac{n^{m-1}}{2^{m-1}(m-1)!}} \\
&\quad \times[e^{\frac{1}{2}\log(\frac{p}{q})}\alpha q + e^{\frac{1}{2}\log(\frac{1-p}{1-q})}\alpha(1-q) + (1-\alpha)]^{\frac{n^{m-1}}{2^{m-1}(m-1)!}}.
\end{aligned}$$

Taking the logarithm of both sides yields

$$\begin{aligned}
\log(I) &\leq \frac{1}{2}r\delta\log n + \frac{n^{m-1}\alpha}{2^{m-1}(m-1)!}[2\sqrt{pq} + 2\sqrt{(1-p)(1-q)} - 2] \\
&= \frac{1}{2}\frac{\log n}{\sqrt{\log(\frac{1}{\delta})}} - \frac{t\log n}{2^{m-1}(m-1)!}[(\sqrt{p}-\sqrt{q})^2 + (\sqrt{1-p}-\sqrt{1-q})^2].
\end{aligned}$$

Hence,

$$(I) \leq n^{-t\frac{(\sqrt{p}-\sqrt{q})^2+(\sqrt{1-p}-\sqrt{1-q})^2}{2^{m-1}(m-1)!}(1+o(1))} = n^{-\frac{t}{I_m(p,q)}(1+o(1))}.$$

Because $t > I_m(p,q)$, by assumption, we have $(I) \leq n^{-(1+\epsilon)}$, for some small constant $\epsilon > 0$, and hence

$$p_i \leq (I) + (II) \leq n^{-(1+\epsilon)}.$$

By the union bound, the probability that a mislabeled node exists is bounded by $n^{-\epsilon} = o(1)$. The proof is complete.

3.4. Proof of Theorem 2.4

Recall $\widetilde{A}_{i_1 i_2 \ldots i_m} = \mathbb{1}[A_{i_1 i_2 \ldots i_m} = 1]$. For convenience, let $e = \{i_1, i_2, \ldots, i_m\}$ for distinct nodes $i_1, i_2, \ldots, i_m$. Let $p_1 = p\alpha$ and $q_1 = q\alpha$. Then

$$\mathbb{E}(\widetilde{A}_e) = \begin{cases} p_1 & e \subset I_+(\sigma), or\ I_-(\sigma); \\ q_1, & otherwise. \end{cases}$$

$$Var(\widetilde{A}_e) = \begin{cases} p_1(1-p_1), & e \subset I_+(\sigma), or\ I_-(\sigma); \\ q_1(1-q_1), & otherwise. \end{cases}$$

Let $\mathbb{I}_e$ denote an n-dimensional vector with i_l-th position one, $i_l \in e$, $l = 1, 2, \ldots, m$, and other positions are zero. Denote $\sigma_e = diag(\sigma) \cdot \mathbb{I}_e$ and

$$L = \sum_e \widetilde{A}_e \left[(\mathbb{I}_e^\intercal \sigma_e) diag(\sigma_e) - \sigma_e \sigma_e^\intercal \right].$$

Let I be the identity matrix and $M = I - \frac{II^T}{n} - \frac{\sigma\sigma^T}{n}$. By Proposition 2 in Kim et al. (2018), it suffices to show the third smallest eigenvalue of $M(\mathbb{E}(L))M$ is larger than zero with probability $1 - o(1)$, that is, $\lambda_3 \left(M\mathbb{E}(L)M \right) > 0$.

First, we have the following result,

Proposition 3.4.1. *Let $M = I - \frac{II^T}{n} - \frac{\sigma\sigma^T}{n}$. Then*

$$M(\mathbb{E}(L))M = \frac{p_1 - q_1}{2} n \binom{\frac{n}{2} - 2}{m-2} M,$$

and

$$\lambda_3 \left(M\mathbb{E}(L)M \right) = \frac{p_1 - q_1}{2^{m-1}} \frac{n^{m-1}}{(m-2)!} \left(1 + o(1) \right).$$

Proof. Simple calculation yields

$$\begin{aligned} \mathbb{E}L &= \frac{< \mathbb{E}L, M >}{n-2} M + \frac{< \mathbb{E}L, \sigma\sigma^\intercal >}{n^2} \sigma\sigma^\intercal \\ < \mathbb{E}L, \sigma\sigma^\intercal > &= -q_1 n^2 \binom{n-2}{m-2} \\ < \mathbb{E}L, M > &= \frac{p_1 - q_1}{2} n \binom{n}{n-2} \binom{n/2 - 2}{m-2}. \end{aligned}$$

Hence, $M(\mathbb{E}(L))M = \frac{< \mathbb{E}L, M >}{n-2} M$. The proof is complete. □

Next, we present the Matrix Bernstein inequality.

Lemma 3.4.2 (Matrix Bernstein inequality)**.** *Let $\{X_k\}$ be a finite sequence of independent, symmetric random matrices of dimension N. Suppose that $\mathbb{E}X_k = 0$ and $\|X_k\| \leq M$ almost surely, for*

all k. Then, for all $x \geq 0$,

$$\mathbb{P}(\|\sum_k X_k\| \geq x) \leq N \cdot \exp\left(-\frac{\frac{x^2}{2}}{v^2 + \frac{Mx}{3}}\right),$$

where $v^2 = \|\sum_k \mathbb{E}X_k^2\|$.

Recall that

$$L = \sum_e \widetilde{A}_e((\mathbb{I}_e^\intercal \sigma_e) diag(\sigma_e) - \sigma_e \sigma_e^\intercal).$$

Hence,

$$\Pi(L - \mathbb{E}L)\Pi = \sum_e (\widetilde{A}_e - \mathbb{E}(\widetilde{A}_e) \cdot \Pi((\mathbb{I}_e^\intercal \sigma_e) diag(\sigma_e) - \sigma_e \sigma_e^\intercal)\Pi.$$

We note that

$$\|\Pi((\mathbb{I}_e^\intercal \sigma_e) diag(\sigma_e) - \sigma_e \sigma_e^\intercal)\Pi\| \leq |\mathbb{I}_e^\intercal \sigma_e| + \|\sigma_e\|^2 \leq 2m$$

for any hyperdge e. By Lemma 3.4.2, we have

$$\mathbb{P}(\|\Pi(L - \mathbb{E}L)\Pi\| \geq x) \leq n \cdot \exp\left(-\frac{\frac{x^2}{2}}{v^2 + \frac{2mx}{3}}\right)$$

where

$$v^2 = \left\| \sum_e \mathbb{E}((A_{\mathcal{H}})_e - \mathbb{E}(A_{\mathcal{H}})_e)^2 (\cdot \Pi((\mathbb{I}_e^\intercal \sigma_e) diag(\sigma_e) - \sigma_e \sigma_e^\intercal)\Pi)^2 \right\|.$$

Let $v^2 = \|\sum_e \mathbb{E}(\widetilde{A}_e - \mathbb{E}(\widetilde{A}_e))^2 [M((\mathbb{I}_e^\intercal \sigma_e) diag(\sigma_e) - \sigma_e \sigma_e^\intercal) M]^2\|$, $V = \sum_e \mathbb{E}(\widetilde{A}_e - \mathbb{E}(\widetilde{A}_e))^2 Y_e$ and

$$Y_e = [(\mathbb{I}_e^\intercal \sigma_e) diag(\sigma_e) - \sigma_e \sigma_e^\intercal)] M [(\mathbb{I}_e^\intercal \sigma_e) diag(\sigma_e) - \sigma_e \sigma_e^\intercal)].$$

The following result gives the expression of V and v^2.

Proposition 3.4.3. $V = t_1 \frac{1}{n} \sigma \sigma^\intercal + t_2 M$, *where*

$$t_1 = \frac{m n^{m-1}}{(m-2)!} q_2, \qquad t_2 = \frac{n^{m-1}}{(m-2)!} [\frac{m}{2^{m-1}} p_2 - \frac{m - 2^m}{2^{m-1}} q_2],$$

and $v^2 = \|MVM\| = \|t_2 M\| = t_2(1 + o(1))$.

Proof. Note that $v^2 = \|\sum_e \mathbb{E}(\widetilde{A}_e - \mathbb{E}(\widetilde{A}_e))^2 M Y_e M\| = \|MVM\|$, we need to show that $V = t_1 \frac{1}{n}\sigma\sigma^\intercal + t_2 M$. Note that $V = \sum_{e\in \mathbb{1}_+(\sigma), \mathbb{1}_-(\sigma)} p_1(1-p_1)Y_e + \sum_{\substack{e\notin \mathbb{1}_+ \\ e\notin \mathbb{1}_-}} q_1(1-q_1)Y_e$. It is clear that

$$V = \frac{< V, M >}{< M, M >} M + < V, \frac{1}{n}\sigma\sigma^\intercal > \frac{1}{n}\sigma\sigma^\intercal.$$

Hence, $t_1 = \frac{1}{n} < V, \sigma\sigma^\intercal >, t_2 = \frac{1}{n-2} < V, M >= \frac{1}{n-2}[tr(V) - \frac{\sigma M \sigma^\intercal}{n}]$. Direct calculation yields

$$t_1 = \frac{mn^{m-1}}{(m-2)!}q_2, \quad t_2 = \frac{n^{m-1}}{(m-2)!}\left[\frac{m}{2^{m-1}}p_2 - \frac{m-2^m}{2^{m-1}}q_2\right].$$

□

Proof of Theorem 2.4. Note that

$$M(L - \mathbb{E}(L))M = \sum_e [(\widetilde{A}e - \mathbb{E}(\widetilde{A}e))]M((\mathbb{I}_e^\intercal \sigma_e) diag(\sigma_e) - \sigma_e\sigma_e^\intercal))M.$$

By Lemma 3.4.2, let $x = (1+\epsilon)\sqrt{2\log n}v$,

$$\begin{aligned} \mathbb{P}(\|M(L - \mathbb{E}(L)M\|) \geq t) &\leq ne^{-\frac{\frac{(1+L)^2 2\log n}{2} v^2}{v^2 + \frac{4m(1+\epsilon)\sqrt{2\log n}v}{3}}} = ne^{-\frac{(1+\epsilon)^2 \log n}{1 + \frac{4m(1+\epsilon)\sqrt{2\log n}}{3v}}} \\ &\leq e^{-\log n\left\{\frac{(1+\epsilon)^2}{1+\frac{4m(1+\epsilon)\sqrt{2\log n}}{3v}} - 1\right\}}. \end{aligned}$$

If $\frac{4m\sqrt{2\log n}}{3v} \leq 1$, that is,

$$t \geq \frac{32m^2(m-2)!\,2^{m-1}}{9(mp - (m-2^m)q)}, \tag{3.3}$$

take $\epsilon = 1$. If $\frac{4m\sqrt{2\log n}}{3v} > 1$, take $1 + \epsilon = \delta\frac{8m\sqrt{2\log n}}{3v}$ for any constant $\delta > 1$. Either case, we have

$$\mathbb{P}(\|M(L - \mathbb{E}(L)M\|) \geq t) = o(1).$$

With probability $1 - o(1)$, we have

$$\|M(L - \mathbb{E}(L)M\| < (1+\epsilon)\sqrt{2t_2\log n}.$$

If

$$\frac{(p_1-q_1)}{2^{m-1}}\frac{n^{m-1}}{(m-2)!} > (1+\epsilon)\sqrt{2t_2\log n}, \tag{3.4}$$

then, $\lambda_3(M\mathbb{E}(L)M) > 0$ with probability $1-o(1)$.

When (3.3) holds, $\epsilon = 1$ and (3.4) is equivalent to

$$\begin{aligned}
[\frac{(p_1-q_1)}{2^{m-1}}\frac{n^{m-1}}{(m-2)!}]^2 &> (1+\epsilon)^2 2\log n\left(\frac{n^{m-1}}{(m-2)!}[\frac{m}{2^{m-1}}p_2 - \frac{m-2^m}{2^{m-1}}q_2]\right),\\
\frac{(p_1-q_1)}{2^{m-1}}\frac{n^{m-1}}{(m-2)!} &> (1+\epsilon)^2 2\log n[mp_2-(m-2^m)q_2],\\
\frac{n^{m-1}(p-q)^2\alpha^2(1-\eta)^2}{2^{m-1}(m-2)!} &> (1+\epsilon)^2 2\log n[m\alpha p-(m-2^m)\alpha q],\\
\frac{n^{m-1}(p-q)^2\alpha}{2^{m-1}(m-2)!} &> (1+\epsilon)^2 2\log n[mp-(m-2^m)q].
\end{aligned}$$

Note that $\alpha = \frac{t\log n}{n^{m-1}}$. Hence,

$$\frac{t(p-q)^2}{2^{m-1}(m-2)!} > 8[mp-(m-2^m)q].$$

Then,

$$t > 8\frac{2^{m-1}(m-2)!\,[mp-(m-2^m)q]}{(p-q)^2} = J_m(p,q),$$

which is larger than the right hand of (3.3).

When (3.3) fails, $1+\epsilon = \delta\frac{8m\sqrt{2\log n}}{3v}$, and (3.4) is equivalent to

$$\frac{(p_1-q_1)}{2^{m-1}}\frac{n^{m-1}}{(m-2)!} > \delta\frac{8m\sqrt{2\log n}}{3}\sqrt{2\log n}, \tag{3.5}$$

hence,

$$t > \delta\frac{16m}{3}\frac{2^{m-1}(m-2)!}{p-q}, \quad \delta > 1,$$

which is not possible. The proof is complete.

4. NUMERICAL SIMULATION

In this section, we conduct numerical simulations to gauge the performance of our proposed algorithms, and evaluate the effectiveness of the imputation methods, we conducted numerical simulations. Furthermore, to demonstrate the process of community detection in hypergraphs with more than two communities, we incorporated the Spectral Hypergraph Partitioning algorithm (Ghoshdastidar and Dukkipati (2014)). We executed corresponding simulations that specifically involved the use of the community density imputation method as an auxiliary tool in the context of multi-community detection. Our objective was to ascertain the robustness of these methodologies and offer a thorough understanding of their practical implementation.

Part of this work used resources from the Center for Computationally Assisted Science and Technology (CCAST) at North Dakota State University, which were made possible in part by NSF MRI Award No. 2019077. In this context, we deal with an equal-community setting, and we utilize the "error rate" as the evaluation metric. This is intended to accurately represent the performance of both the proposed algorithms and the Spectral Hypergraph Partitioning algorithm. This metric will allow us to quantitatively assess the precision and reliability of these algorithms in the community detection task.

The error rate is computed as the ratio of incorrectly clustered instances to the total number of instances in the dataset. In an ideal scenario, the error rate would be 0.0, indicating a perfect clustering. Conversely, a higher error rate signifies poorer clustering. The error rate can be calculated using the following formula:

$$\ell(\hat{\sigma}, \sigma) = \min_{\pi \in S_k} \frac{1}{n} \sum_{\mu \in [n]} \mathbb{1}[\hat{\sigma}(\mu) \neq \pi(\sigma(\mu))] \tag{4.1}$$

where S_k stands for the symmetric group on $[k]$, which consists of all permutations of $[k]$. Here, k is the number of communities, $\hat{\sigma}$ is the predicted label, and σ is the true label of node μ. The indicator function counts the proportion of misclassified nodes between an estimator and the ground truth assignment. Considering the issue of possible relabeling, the error rate is defined as the loss function,

which maximizes the agreement between an estimator and the ground truth after an alignment by label permutation.

We employed four distinct node configurations, specifically n=50, n=100, n=150, and n=200. Additionally, we utilized four different reveal rate (alpha) settings for each of the two algorithms. For the Efficient Spectral algorithm, these were $\alpha = 0.9$, $\alpha = 0.8$, $\alpha = 0.7$, and $\alpha = 0.6$. For the Semi-definite programming algorithm, these were $\alpha = 0.013$, $\alpha = 0.12$, $\alpha = 0.011$, and $\alpha = 0.010$. We also set the pair of in-community and cross-community probabilities p and q as $p = 0.425, q = 0.075$, $p = 0.400, q = 0.100$, $p = 0.375, q = 0.125$, and $p = 0.350, q = 0.150$ for both algorithms. For each combination of these settings, we conducted 50 repetitions of the simulation.

4.1. Simulation of the Efficient Spectral Algorithm

4.1.1. When p=0.425, q=0.075

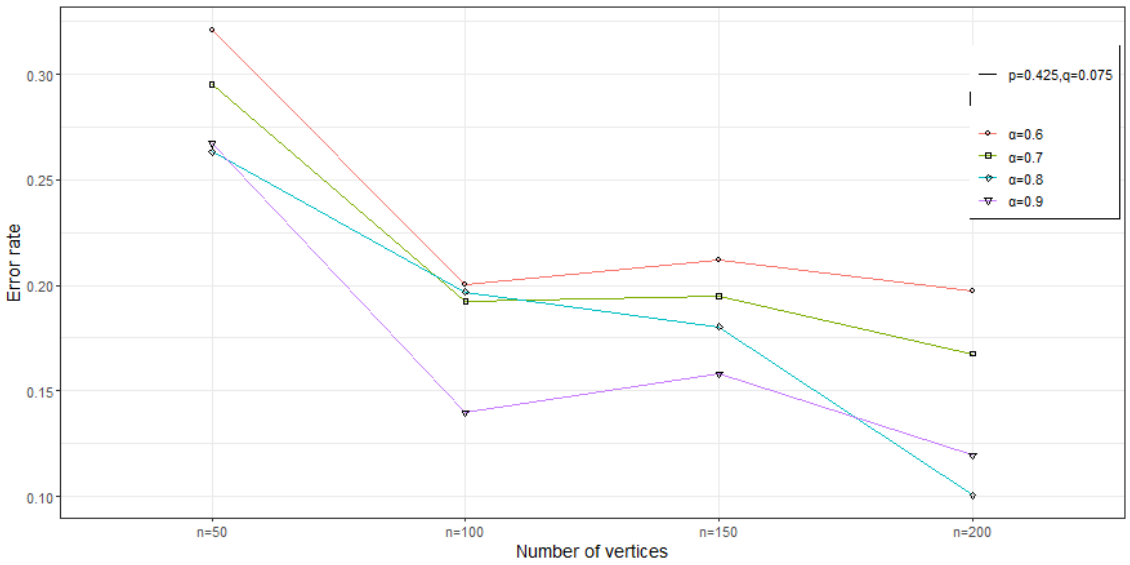

Figure 4.1. Comparison of error rate from Efficient Spectral algorithm at p=0.425, q=0.075. Note: p is the in-community connectivity probability, indicating the probability that vertices in the same community connect with each other; and q is the cross-community connectivity probability, indicating the probability that vertices in different communities connect with each other; α is the reveal rate, indicating the probability that if a hyperedge is observed or not. Same below.

In Figure 4.1, we can observe a clear decreasing trend in the error rate as the number of nodes n increases, while keeping the in-community and cross-community probabilities p and q fixed. Although there are observable fluctuations, the general downward trend is consistent. Moreover, as

the reveal rate α increases (represented by lines of different colors), the error rate correspondingly decreases, which aligns with our expectations.

4.1.2. When p=0.400, q=0.100

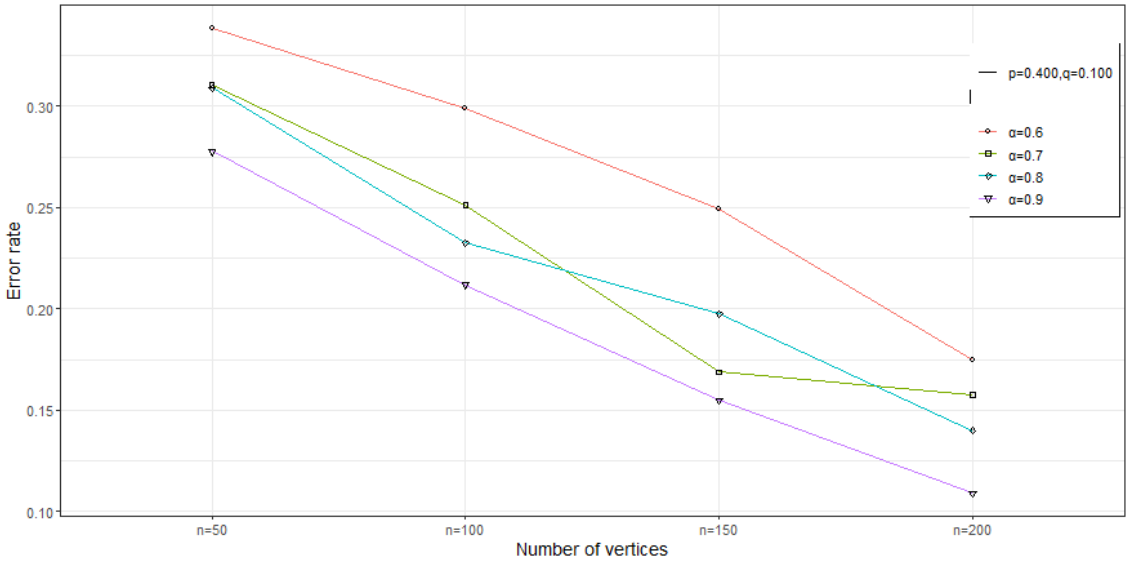

Figure 4.2. Comparison of error rate from Efficient Spectral algorithm at p=0.400, q=0.100

In Figure 4.2, akin to our observations from Figure 4.1, we identify a pronounced decrease in the error rate as the number of nodes n increases, while holding p and q constant. However, a slight deviation from Figure 4.1 is noticed here: the decrease in error rate as the reveal rate α increases is more straightforward and less prone to fluctuations.

4.1.3. When p=0.375, q=0.125

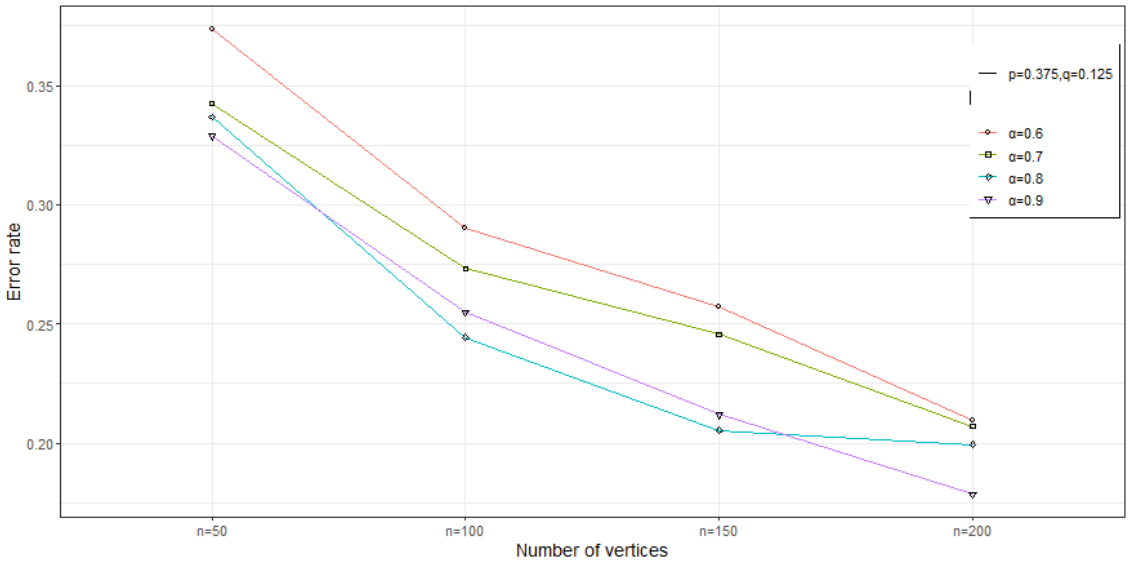

Figure 4.3. Comparison of error rate from Efficient Spectral algorithm at p=0.375, q=0.125

In Figure 4.3, we observe near-perfect results for all combinations of parameters, barring the line corresponding to $\alpha = 0.8$. Although there are some outliers, the overarching trend aligns well with the patterns noted in the previous figures. This suggests that, barring some specific conditions, the decrease in error rate as n and α increase remains consistent.

4.1.4. When p=0.350, q=0.150

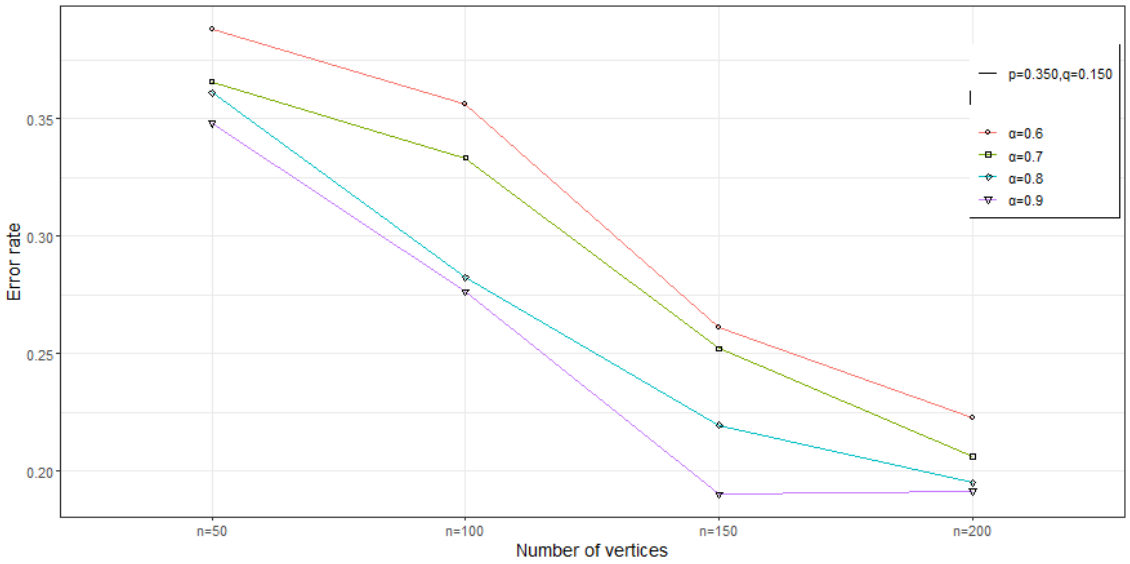

Figure 4.4. Comparison of error rate from Efficient Spectral algorithm at p=0.350, q=0.150

The observations in Figure 4.4 are undoubtedly in line with our expectations. As in the previous figures, the error rate decreases as both n and α increase, reaffirming the consistent trend we have identified across all our simulations. This consistency bolsters our confidence in the effectiveness of the Efficient Spectral algorithm under a variety of parameter settings.

4.1.5. Summary of Efficient Spectral Algorithm

Table 4.1. Error rates of Efficient Spectral algorithm simulation

vertex	alpha	p, q	error_rate	std
$n = 50$	α=0.9	$p = 0.425, q = 0.075$	0.267	0.146
$n = 50$	α=0.8	$p = 0.425, q = 0.075$	0.263	0.133
$n = 50$	α=0.7	$p = 0.425, q = 0.075$	0.295	0.135
$n = 50$	α=0.6	$p = 0.425, q = 0.075$	0.321	0.130
$n = 100$	α=0.9	$p = 0.425, q = 0.075$	0.140	0.137
$n = 100$	α=0.8	$p = 0.425, q = 0.075$	0.197	0.161

Error rates of Efficient Spectral algorithm simulation (continued)

vertex	alpha	p, q	error_rate	std
$n = 100$	α=0.7	$p = 0.425, q = 0.075$	0.192	0.163
$n = 100$	α=0.6	$p = 0.425, q = 0.075$	0.200	0.156
$n = 150$	α=0.9	$p = 0.425, q = 0.075$	0.158	0.154
$n = 150$	α=0.8	$p = 0.425, q = 0.075$	0.180	0.158
$n = 150$	α=0.7	$p = 0.425, q = 0.075$	0.195	0.139
$n = 150$	α=0.6	$p = 0.425, q = 0.075$	0.212	0.127
$n = 200$	α=0.9	$p = 0.425, q = 0.075$	0.120	0.161
$n = 200$	α=0.8	$p = 0.425, q = 0.075$	0.101	0.159
$n = 200$	α=0.7	$p = 0.425, q = 0.075$	0.168	0.165
$n = 200$	α=0.6	$p = 0.425, q = 0.075$	0.198	0.179
$n = 50$	α=0.9	$p = 0.400, q = 0.100$	0.278	0.140
$n = 50$	α=0.8	$p = 0.400, q = 0.100$	0.309	0.137
$n = 50$	α=0.7	$p = 0.400, q = 0.100$	0.310	0.125
$n = 50$	α=0.6	$p = 0.400, q = 0.100$	0.338	0.128
$n = 100$	α=0.9	$p = 0.400, q = 0.100$	0.212	0.170
$n = 100$	α=0.8	$p = 0.400, q = 0.100$	0.232	0.156
$n = 100$	α=0.7	$p = 0.400, q = 0.100$	0.251	0.182
$n = 100$	α=0.6	$p = 0.400, q = 0.100$	0.299	0.150
$n = 150$	α=0.9	$p = 0.400, q = 0.100$	0.155	0.149
$n = 150$	α=0.8	$p = 0.400, q = 0.100$	0.198	0.147
$n = 150$	α=0.7	$p = 0.400, q = 0.100$	0.169	0.150
$n = 150$	α=0.6	$p = 0.400, q = 0.100$	0.249	0.153
$n = 200$	α=0.9	$p = 0.400, q = 0.100$	0.109	0.122
$n = 200$	α=0.8	$p = 0.400, q = 0.100$	0.140	0.149
$n = 200$	α=0.7	$p = 0.400, q = 0.100$	0.158	0.157
$n = 200$	α=0.6	$p = 0.400, q = 0.100$	0.174	0.164

Error rates of Efficient Spectral algorithm simulation (continued)

vertex	alpha	p, q	error_rate	std
$n = 50$	α=0.9	$p = 0.375, q = 0.125$	0.329	0.123
$n = 50$	α=0.8	$p = 0.375, q = 0.125$	0.337	0.117
$n = 50$	α=0.7	$p = 0.375, q = 0.125$	0.342	0.121
$n = 50$	α=0.6	$p = 0.375, q = 0.125$	0.374	0.094
$n = 100$	α=0.9	$p = 0.375, q = 0.125$	0.255	0.153
$n = 100$	α=0.8	$p = 0.375, q = 0.125$	0.244	0.141
$n = 100$	α=0.7	$p = 0.375, q = 0.125$	0.273	0.135
$n = 100$	α=0.6	$p = 0.375, q = 0.125$	0.290	0.139
$n = 150$	α=0.9	$p = 0.375, q = 0.125$	0.212	0.129
$n = 150$	α=0.8	$p = 0.375, q = 0.125$	0.205	0.141
$n = 150$	α=0.7	$p = 0.375, q = 0.125$	0.246	0.155
$n = 150$	α=0.6	$p = 0.375, q = 0.125$	0.257	0.146
$n = 200$	α=0.9	$p = 0.375, q = 0.125$	0.179	0.164
$n = 200$	α=0.8	$p = 0.375, q = 0.125$	0.200	0.155
$n = 200$	α=0.7	$p = 0.375, q = 0.125$	0.207	0.177
$n = 200$	α=0.6	$p = 0.375, q = 0.125$	0.210	0.183
$n = 50$	α=0.9	$p = 0.350, q = 0.150$	0.348	0.114
$n = 50$	α=0.8	$p = 0.350, q = 0.150$	0.361	0.108
$n = 50$	α=0.7	$p = 0.350, q = 0.150$	0.366	0.099
$n = 50$	α=0.6	$p = 0.350, q = 0.150$	0.388	0.095
$n = 100$	α=0.9	$p = 0.350, q = 0.150$	0.276	0.128
$n = 100$	α=0.8	$p = 0.350, q = 0.150$	0.282	0.125
$n = 100$	α=0.7	$p = 0.350, q = 0.150$	0.333	0.125
$n = 100$	α=0.6	$p = 0.350, q = 0.150$	0.356	0.112
$n = 150$	α=0.9	$p = 0.350, q = 0.150$	0.190	0.136
$n = 150$	α=0.8	$p = 0.350, q = 0.150$	0.219	0.136

Error rates of Efficient Spectral algorithm simulation (continued)

vertex	alpha	p, q	error_rate	std
$n = 150$	α=0.7	$p = 0.350, q = 0.150$	0.252	0.143
$n = 150$	α=0.6	$p = 0.350, q = 0.150$	0.261	0.142
$n = 200$	α=0.9	$p = 0.350, q = 0.150$	0.192	0.145
$n = 200$	α=0.8	$p = 0.350, q = 0.150$	0.195	0.161
$n = 200$	α=0.7	$p = 0.350, q = 0.150$	0.206	0.173
$n = 200$	α=0.6	$p = 0.350, q = 0.150$	0.223	0.154

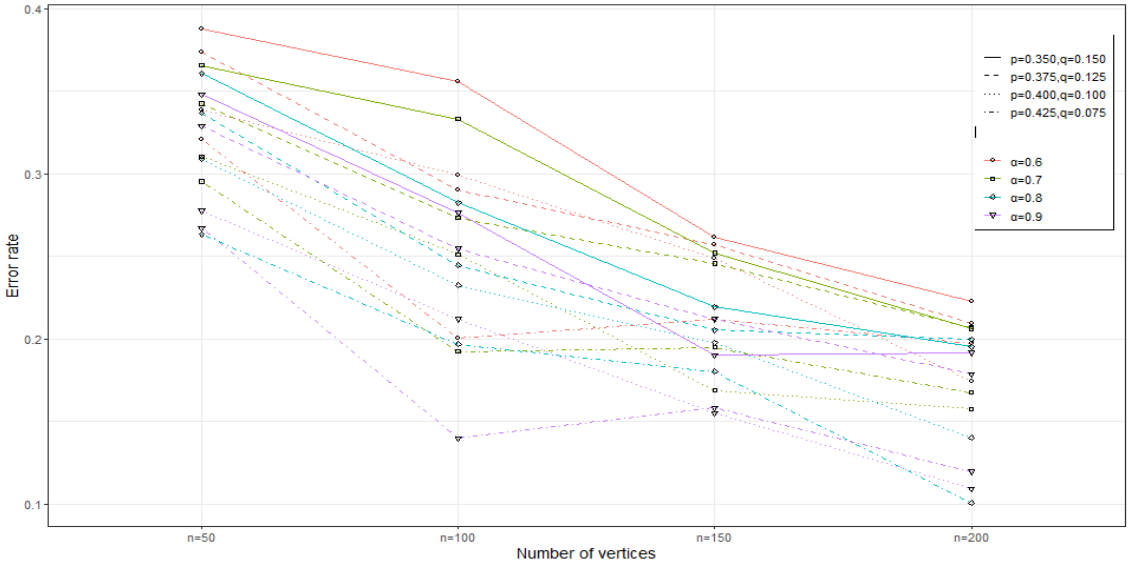

Figure 4.5. Comparison of error rate from Efficient Spectral algorithm at all settings

After synthesizing all the aforementioned figures, the decreasing trend remains prominently discernible in Figure 4.5. This serves as a numerical substantiation of Theorem 2.3. The observed pattern indicates that as the number of nodes and the reveal rate increase, the error rate reduces—showing the robustness of our proposed algorithms over a variety of parameter settings, ultimately reinforcing the veracity of our theoretical findings.

4.2. Simulation of the Semi-Definite Programming Algorithm

4.2.1. When p=0.425, q=0.075

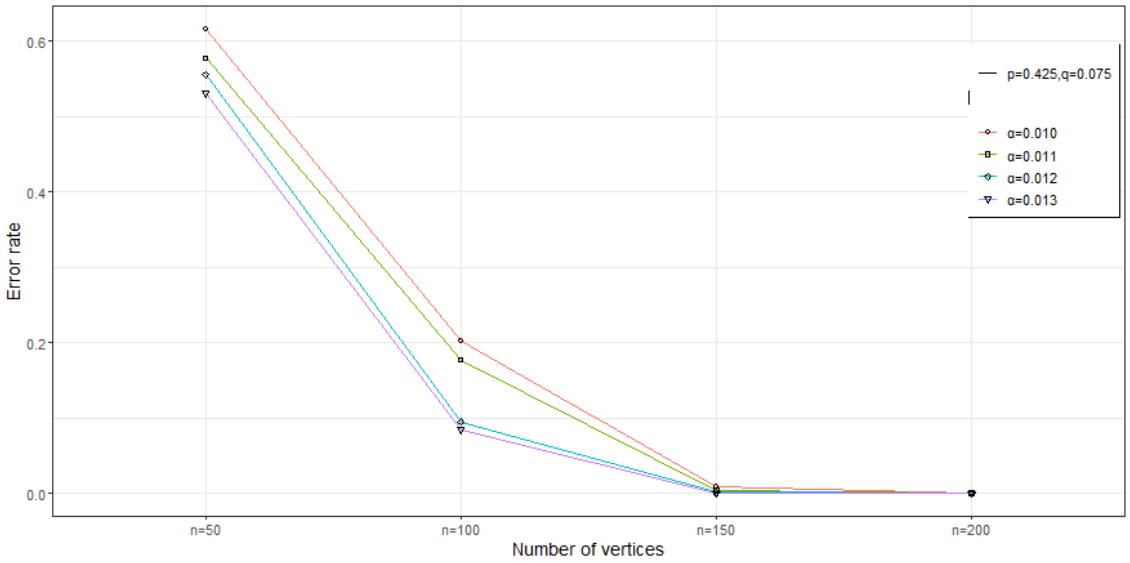

Figure 4.6. Comparison of error rate from Semi-Definite Programming algorithm at p=0.425, q=0.075

In contrast to the Efficient Spectral algorithm, Figure 4.6 exhibits a steep decline as n increases. As anticipated, with p and q held constant, the error rate diminishes significantly as n expands and as α escalates. This observation corroborates our initial expectations, further showcasing the efficacy of our proposed Semi-Definite Programming algorithm in the face of increasing hypergraph size and reveal rates.

4.2.2. When p=0.400, q=0.100

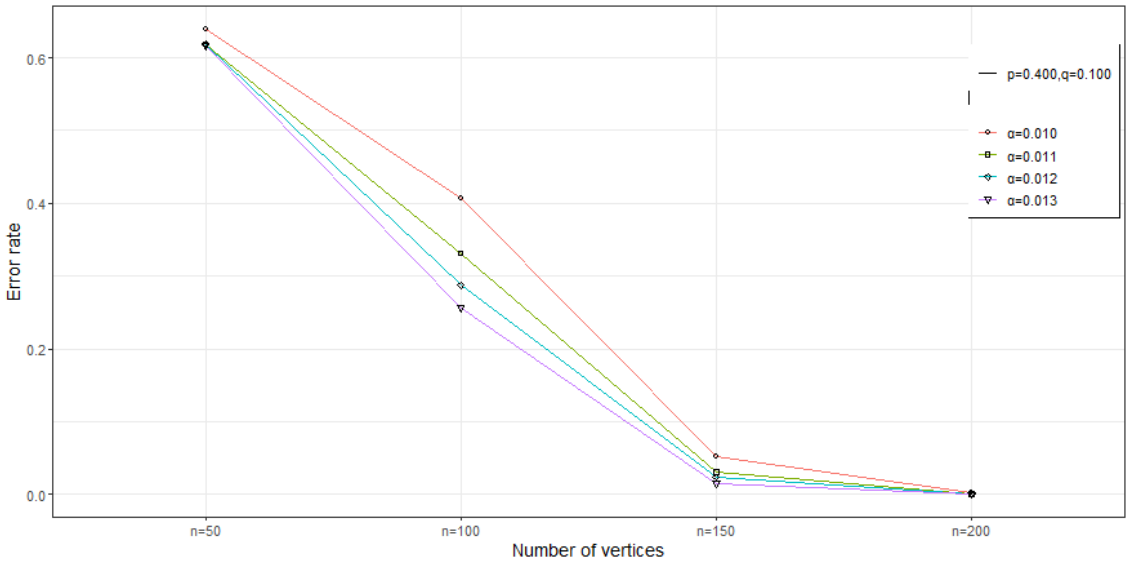

Figure 4.7. Comparison of error rate from Semi-Definite Programming algorithm at p=0.400, q=0.100

The pattern of decline persists in Figure 4.7, consistent with our prior observations. There are no significant outliers, thus affirming the robustness of our Semi-Definite Programming algorithm under various settings. The effectiveness of the algorithm appears consistent, irrespective of the hypergraph size and the chosen reveal rates.

4.2.3. When p=0.375, q=0.125

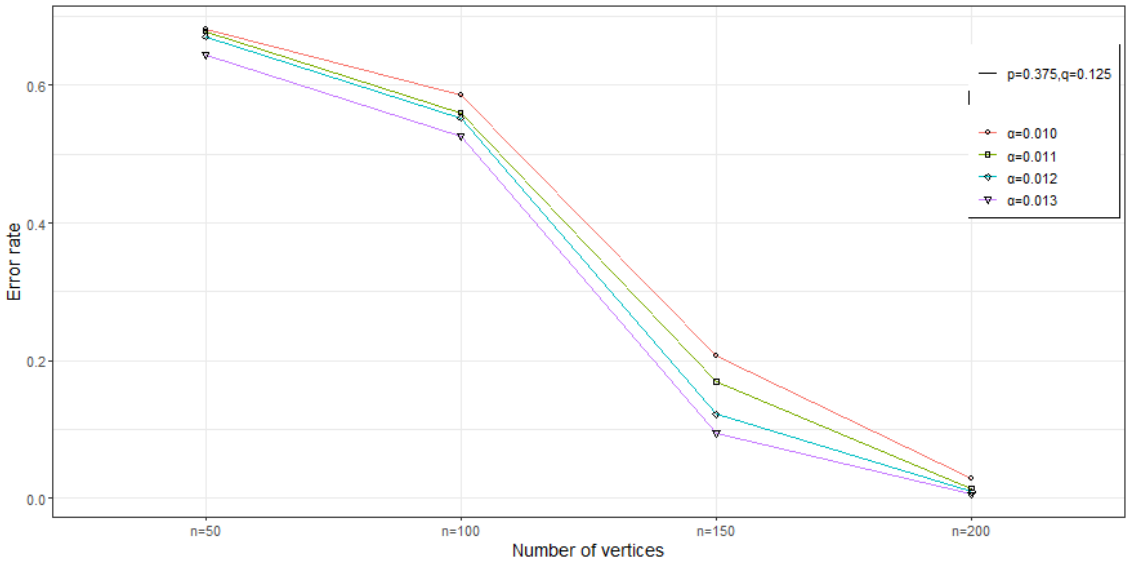

Figure 4.8. Comparison of error rate from Semi-Definite Programming algorithm at p=0.375, q=0.125

The trend of reduction remains consistent in Figure 4.8, similar to the patterns observed previously. There are no substantial outliers, reinforcing the stability of our Semi-Definite Programming algorithm under different scenarios. This demonstrates the robustness of our algorithm irrespective of the hypergraph size or the reveal rates selected.

4.2.4. When p=0.350, q=0.150

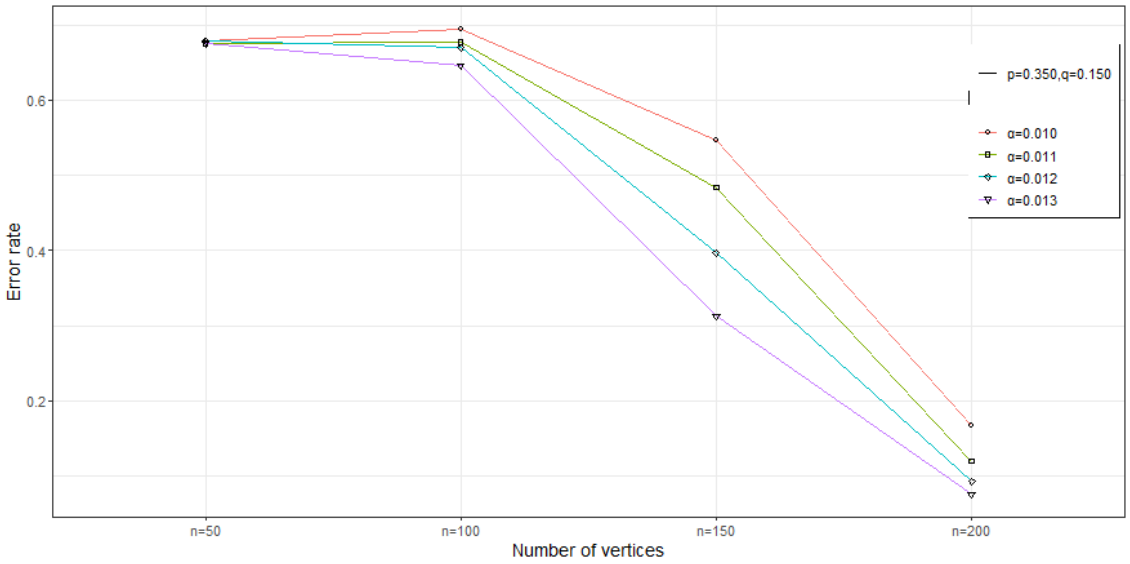

Figure 4.9. Comparison of error rate from Semi-Definite Programming algorithm at p=0.350, q=0.150

As observed in previous figures, Figure 4.9 continues to showcase a consistent decrease in the error rate, further validating the effectiveness of our Semi-Definite Programming algorithm. The absence of significant outliers underscores the reliability of our algorithm across varying hypergraph sizes and reveal rates. These results align with our expectations and demonstrate the robustness of the algorithm across different parameter settings.

4.2.5. Summary of Semi-Definite Programming Algorithm

Table 4.2. Error rates of Semi-Definite Programming algorithm simulation

vertex	alpha	p, q	error_rate	std
$n = 50$	α=0.013	$p = 0.425, q = 0.075$	0.531	0.132
$n = 50$	α=0.012	$p = 0.425, q = 0.075$	0.555	0.113
$n = 50$	α=0.011	$p = 0.425, q = 0.075$	0.578	0.094
$n = 50$	α=0.010	$p = 0.425, q = 0.075$	0.616	0.086
$n = 100$	α=0.013	$p = 0.425, q = 0.075$	0.085	0.048

Error rates of Semi-Definite Programming algorithm simulation (continued)

vertex	alpha	p, q	error_rate	std
$n = 100$	α=0.012	$p = 0.425, q = 0.075$	0.095	0.040
$n = 100$	α=0.011	$p = 0.425, q = 0.075$	0.177	0.076
$n = 100$	α=0.010	$p = 0.425, q = 0.075$	0.202	0.090
$n = 150$	α=0.013	$p = 0.425, q = 0.075$	0	0
$n = 150$	α=0.012	$p = 0.425, q = 0.075$	0.002	0.008
$n = 150$	α=0.011	$p = 0.425, q = 0.075$	0.004	0.011
$n = 150$	α=0.010	$p = 0.425, q = 0.075$	0.009	0.013
$n = 200$	α=0.013	$p = 0.425, q = 0.075$	0	0
$n = 200$	α=0.012	$p = 0.425, q = 0.075$	0	0
$n = 200$	α=0.011	$p = 0.425, q = 0.075$	0	0.003
$n = 200$	α=0.010	$p = 0.425, q = 0.075$	0	0
$n = 50$	α=0.013	$p = 0.400, q = 0.100$	0.616	0.086
$n = 50$	α=0.012	$p = 0.400, q = 0.100$	0.618	0.090
$n = 50$	α=0.011	$p = 0.400, q = 0.100$	0.619	0.065
$n = 50$	α=0.010	$p = 0.400, q = 0.100$	0.639	0.060
$n = 100$	α=0.013	$p = 0.400, q = 0.100$	0.257	0.107
$n = 100$	α=0.012	$p = 0.400, q = 0.100$	0.288	0.107
$n = 100$	α=0.011	$p = 0.400, q = 0.100$	0.331	0.109
$n = 100$	α=0.010	$p = 0.400, q = 0.100$	0.407	0.107
$n = 150$	α=0.013	$p = 0.400, q = 0.100$	0.015	0.019
$n = 150$	α=0.012	$p = 0.400, q = 0.100$	0.024	0.020
$n = 150$	α=0.011	$p = 0.400, q = 0.100$	0.031	0.026
$n = 150$	α=0.010	$p = 0.400, q = 0.100$	0.052	0.027
$n = 200$	α=0.013	$p = 0.400, q = 0.100$	0	0
$n = 200$	α=0.012	$p = 0.400, q = 0.100$	0	0
$n = 200$	α=0.011	$p = 0.400, q = 0.100$	0	0.002

Error rates of Semi-Definite Programming algorithm simulation (continued)

vertex	alpha	p, q	error_rate	std
$n = 200$	α=0.010	$p = 0.400, q = 0.100$	0.002	0.005
$n = 50$	α=0.013	$p = 0.375, q = 0.125$	0.645	0.051
$n = 50$	α=0.012	$p = 0.375, q = 0.125$	0.671	0.041
$n = 50$	α=0.011	$p = 0.375, q = 0.125$	0.678	0.048
$n = 50$	α=0.010	$p = 0.375, q = 0.125$	0.681	0.043
$n = 100$	α=0.013	$p = 0.375, q = 0.125$	0.527	0.120
$n = 100$	α=0.012	$p = 0.375, q = 0.125$	0.552	0.082
$n = 100$	α=0.011	$p = 0.375, q = 0.125$	0.560	0.071
$n = 100$	α=0.010	$p = 0.375, q = 0.125$	0.586	0.110
$n = 150$	α=0.013	$p = 0.375, q = 0.125$	0.095	0.032
$n = 150$	α=0.012	$p = 0.375, q = 0.125$	0.122	0.052
$n = 150$	α=0.011	$p = 0.375, q = 0.125$	0.169	0.068
$n = 150$	α=0.010	$p = 0.375, q = 0.125$	0.206	0.080
$n = 200$	α=0.013	$p = 0.375, q = 0.125$	0.006	0.009
$n = 200$	α=0.012	$p = 0.375, q = 0.125$	0.010	0.014
$n = 200$	α=0.011	$p = 0.375, q = 0.125$	0.014	0.012
$n = 200$	α=0.010	$p = 0.375, q = 0.125$	0.029	0.019
$n = 50$	α=0.013	$p = 0.350, q = 0.150$	0.675	0.049
$n = 50$	α=0.012	$p = 0.350, q = 0.150$	0.678	0.036
$n = 50$	α=0.011	$p = 0.350, q = 0.150$	0.675	0.051
$n = 50$	α=0.010	$p = 0.350, q = 0.150$	0.678	0.042
$n = 100$	α=0.013	$p = 0.350, q = 0.150$	0.645	0.085
$n = 100$	α=0.012	$p = 0.350, q = 0.150$	0.670	0.064
$n = 100$	α=0.011	$p = 0.350, q = 0.150$	0.677	0.068
$n = 100$	α=0.010	$p = 0.350, q = 0.150$	0.694	0.052
$n = 150$	α=0.013	$p = 0.350, q = 0.150$	0.313	0.083

Error rates of Semi-Definite Programming algorithm simulation (continued)

vertex	alpha	p, q	error_rate	std
$n = 150$	α=0.012	$p = 0.350, q = 0.150$	0.397	0.101
$n = 150$	α=0.011	$p = 0.350, q = 0.150$	0.484	0.107
$n = 150$	α=0.010	$p = 0.350, q = 0.150$	0.547	0.092
$n = 200$	α=0.013	$p = 0.350, q = 0.150$	0.076	0.034
$n = 200$	α=0.012	$p = 0.350, q = 0.150$	0.093	0.047
$n = 200$	α=0.011	$p = 0.350, q = 0.150$	0.120	0.047
$n = 200$	α=0.010	$p = 0.350, q = 0.150$	0.168	0.051

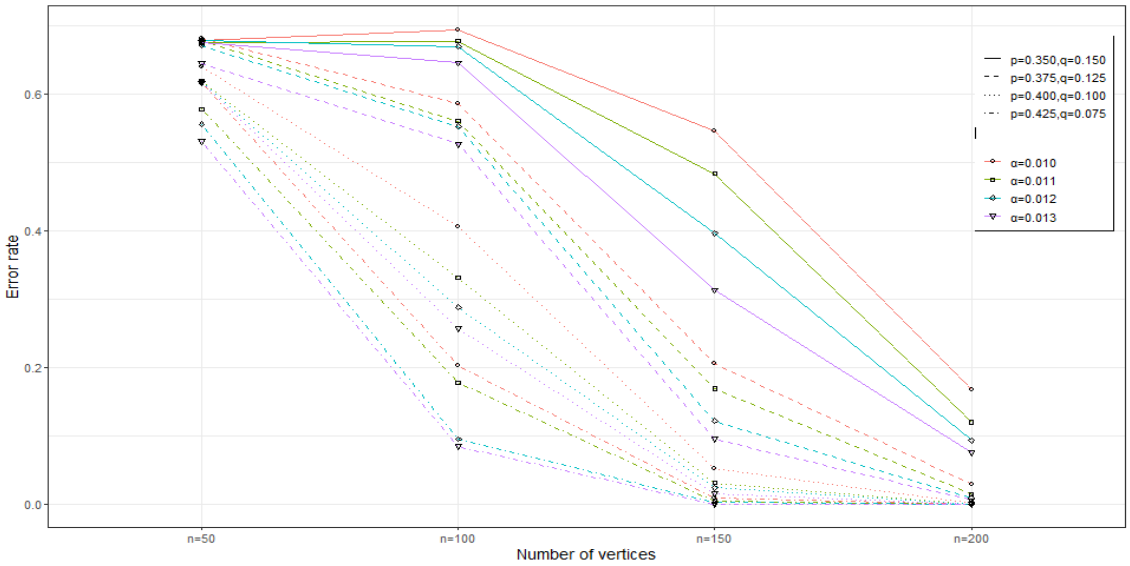

Figure 4.10. Comparison of error rate from Semi-Definite Programming algorithm at all settings

Upon consolidating the figures for the Semi-Definite Programming (SDP) algorithm in Figure 4.10, a pronounced decreasing trend in error rate is readily observable. This pattern aligns precisely with the findings suggested by theorem 2.4. As the size of the hypergraph increases and the reveal rate escalates, the SDP algorithm consistently demonstrates enhanced accuracy, further underscoring the efficacy of our proposed approach.

4.2.6. Comparison of the Two Proposed Algorithms

We thoroughly evaluated the performances of the Efficient Spectral and Semi-Definite Programming (SDP) algorithms through simulations. We calculated the error rates from 50 repeated runs and illustrated the results via a series of figures for each algorithm. The observations highlighted a consistent decrease in the error rate as the number of nodes, n, expanded, while α, p, and q remained constant. When p and q were held constant, the error rate dropped as α escalated for each value of n. This is logical, as α governs the proportion of revealed hyperedges. However, we observed that the Efficient Spectral algorithm necessitated a significantly higher value of α to attain a comparable error rate as the SDP algorithm, given identical values of p, q, and n. This discrepancy in performance suggests that while the Efficient Spectral algorithm is known to achieve a precise detection boundary, it may not be the superior choice or outperform alternative algorithms under all conditions.

4.3. Simulation of the Imputation Methods

In the previous study, we aimed to achieve exact recovery up to the information-theoretic threshold within the framework of CHSBM. To accomplish this, we introduced an efficient spectral algorithm that fulfilled this aim satisfactorily. Nevertheless, how the imputation methods for missing data impact the performance of the proposed algorithms remains undetermined. As a result, we conducted simulations with the proposed imputation methods to assess their impact.

We utilized the Semi-Definite Programming algorithm, introduced in the preceding section, to perform the community detection task for all three imputation methods examined in this study. Regarding the simulation configurations, we used three node settings for all three methods: $n = 50$, $n = 100$, and $n = 150$. For each node setting, three reveal rates (α) and three pairs of in-community connectivity probabilities (p) and cross-community connectivity probabilities (q) were established: $\alpha = 0.05$, $\alpha = 0.04$ and $\alpha = 0.03$, and $p = 0.375, q = 0.125$; $p = 0.350, q = 0.150$; $p = 0.300, q = 0.200$.

The simulation results for each of the methods are presented below.

4.3.1. Simulation Results of the Network Density Method

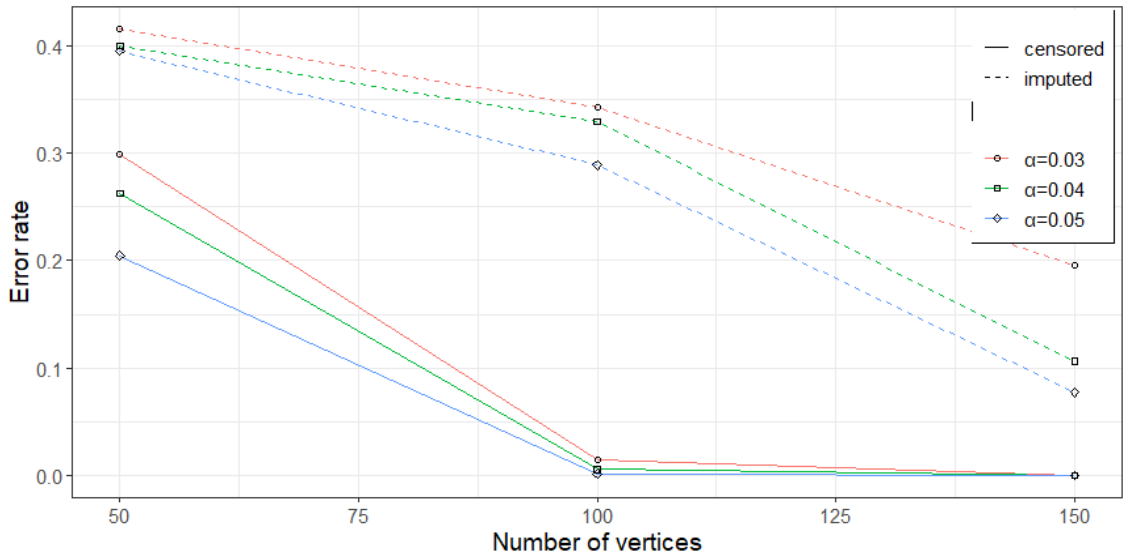

Figure 4.11. Comparison of error rate from censored and imputed hypergraph based on network density, when $p = 0.375$ and $q = 0.125$

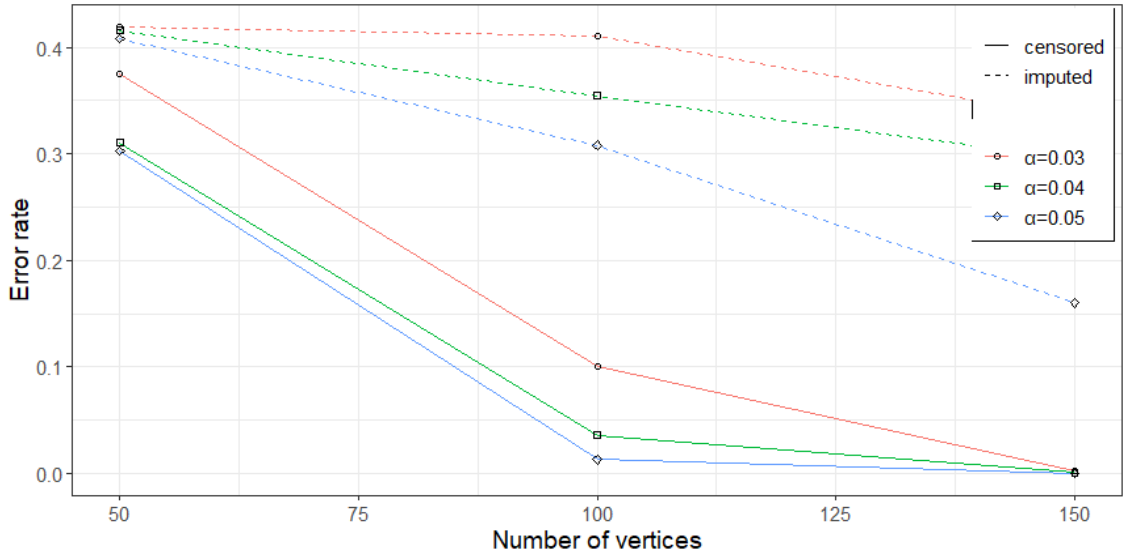

Figure 4.12. Comparison of error rate from censored and imputed hypergraph based on network density, when $p = 0.350$ and $q = 0.150$

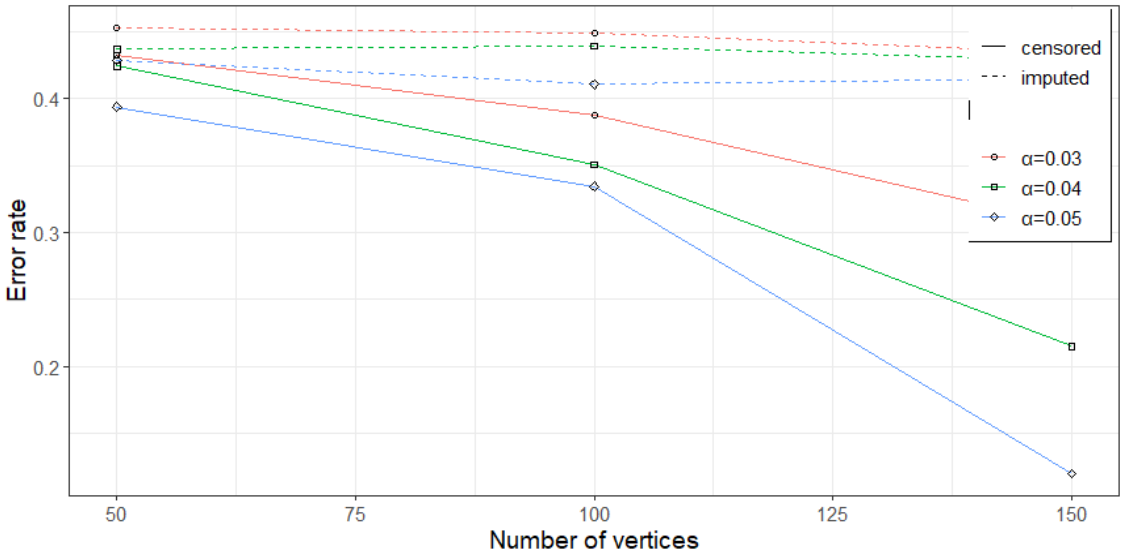

Figure 4.13. Comparison of error rate from censored and imputed hypergraph based on network density, when $p = 0.300$ and $q = 0.200$

In Figures 4.11, 4.12, and 4.13, the solid lines represent the error rates derived from the censored hypergraph data, whereas the dotted lines illustrate the error rates from the data imputed using the network density imputation method. The figures clearly demonstrate that for every combination of p and q, the error rates for the imputed data exhibit an upward trend in comparison to the missing data. This outcome contradicts our initial assumption of a reduction in error rates following the application of the imputation method to the censored hypergraph data, thereby suggesting a potential inadequacy in the underlying rationale of the network density imputation method.

4.3.2. Simulation Results of the Community Density Method

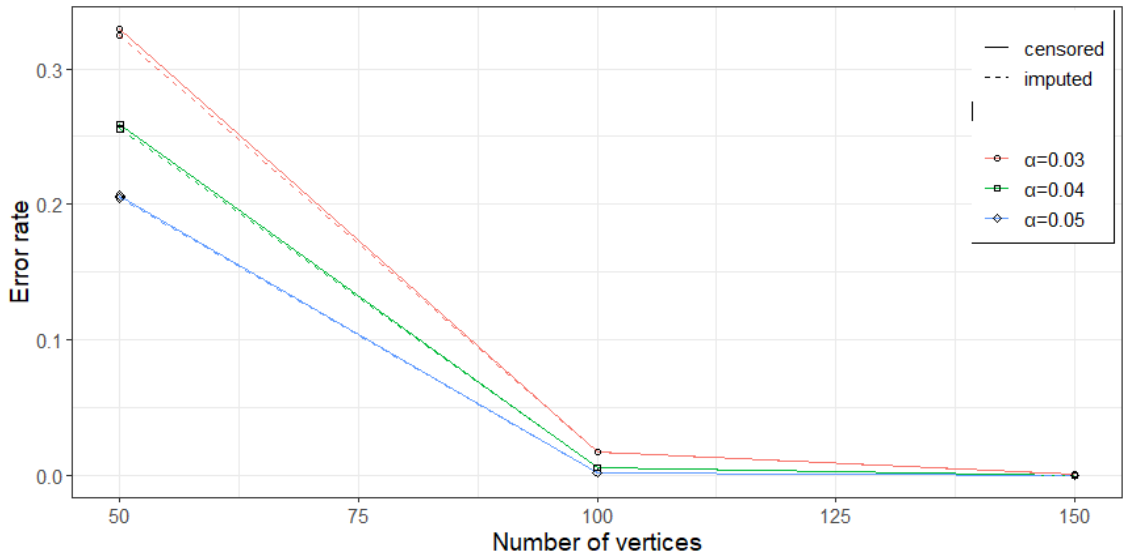

Figure 4.14. Comparison of error rate from censored and imputed hypergraph based on community density, when $p = 0.375$ and $q = 0.125$

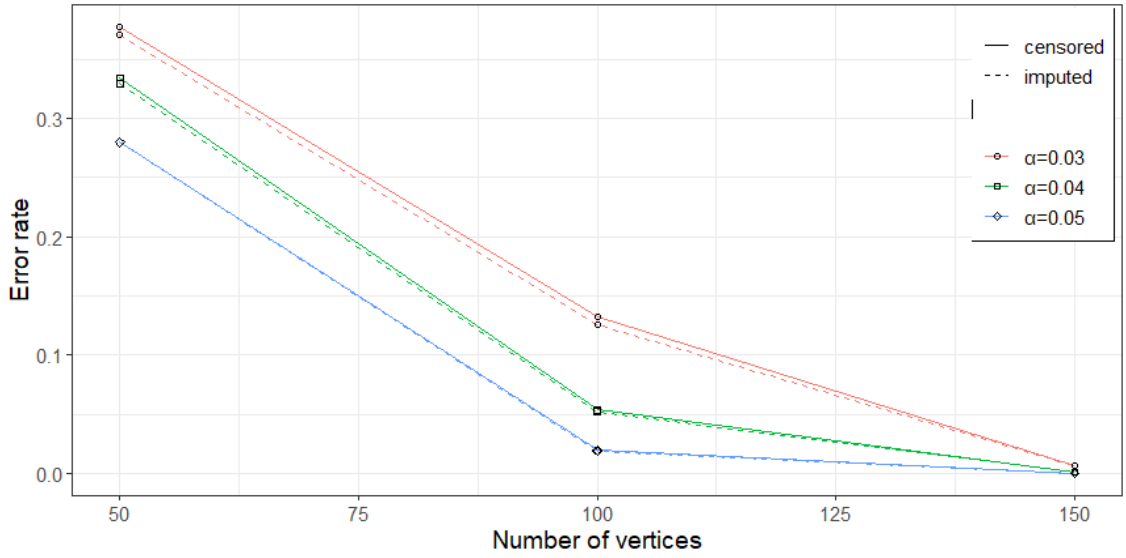

Figure 4.15. Comparison of error rate from censored and imputed hypergraph based on community density, when $p = 0.350$ and $q = 0.150$

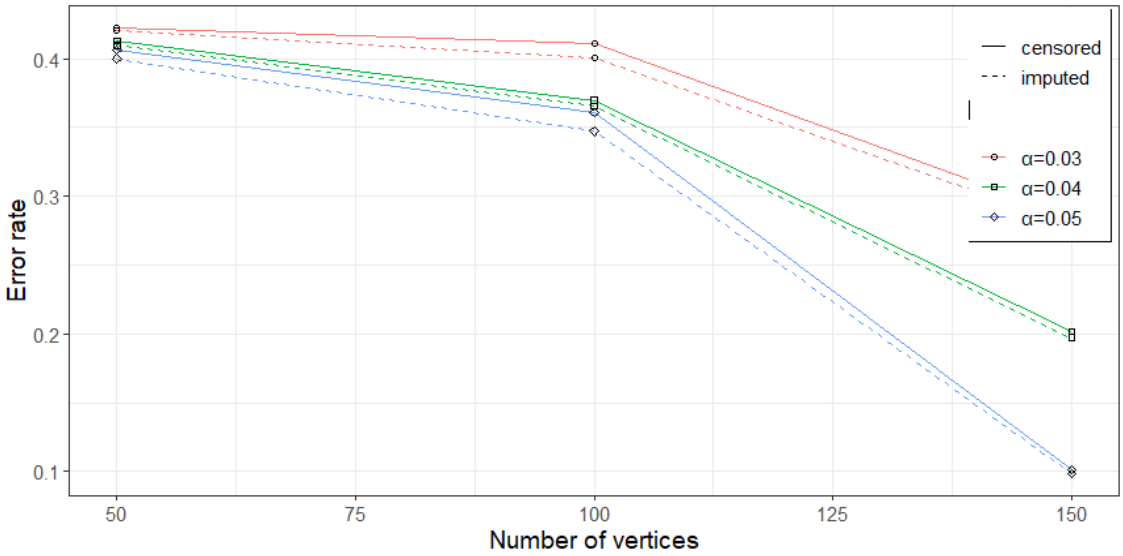

Figure 4.16. Comparison of error rate from censored and imputed hypergraph based on community density, when $p = 0.300$ and $q = 0.200$

Figures 4.14, 4.15, and 4.16 present the error rates from the censored hypergraph data, denoted by the solid lines, compared with those from the data imputed using the community density imputation method, indicated by the dotted lines. With each pair of p and q, the imputed data error rates show a declining trend relative to the missing data. This discrepancy between missing and imputed data becomes more pronounced as the disparity between p and q widens.

To validate that the difference in the error rates between the censored hypergraph and the imputed hypergraph is significant, we performed a paired t-test on the simulation data.

The paired t-test is a powerful statistical tool used to compare the means of two related groups or samples. It is particularly valuable when working with data in which the observations within each group are not independent, such as in before-and-after studies, repeated measures experiments, or matched pairs of observations. This test allows researchers to determine whether there is a statistically significant difference between the means of paired observations.

The paired t-test is based on two hypotheses:

- Null Hypothesis (H_0): There is no significant difference between the means of the paired observations.

- Alternative Hypothesis (H_a): There is a significant difference between the means of the paired observations.

The test statistic (t-value) is calculated by comparing the mean of the paired differences to zero. The formula for the t-value is given by:

$$t = \frac{\text{mean of differences}}{\text{standard error of the differences}}$$

The standard error of the differences takes into account the variability of the differences between the paired observations.

The paired t-test is a robust statistical method used when comparing the means of two related groups. To guarantee its accuracy and reliability, certain assumptions must be met:

1. **Independence of Observations:** Each set of paired observations should be independent from other pairs. In this study, simulations were conducted independently, hence fulfilling this criterion.

2. **Normality of Differences:** The differences between the paired observations should typically follow a normal distribution. Given our sample size of 27 for each data group, we can rely on the Central Limit Theorem. This implies that the t-test remains effective, even with minor deviations from normality.

3. **Homogeneity of Variances:** Variances within each group should be approximately equal. To confirm this, we applied Bartlett's test. With a resultant p-value of 0.941, which is significantly above the 0.05 threshold, the data supports the null hypothesis, suggesting that variances are indeed homogenous across groups.

4. **Interval or Ratio Data:** Data should be at the interval or ratio level. Our data aligns with this requirement as it follows a "before-and-after" pattern.

After ensuring our data meets the prerequisites for a paired t-test, we proceeded with our computations. The resulting p-value is 6.661×10^{-5}, which is considerably below the accepted 0.05 threshold. This strongly contradicts the null hypothesis, suggesting a substantial difference in error

rates between the two groups. Consequently, this indicates that the community density imputation method is noticeably more effective than the network density method.

4.3.3. Simulation Results of the Degree Method

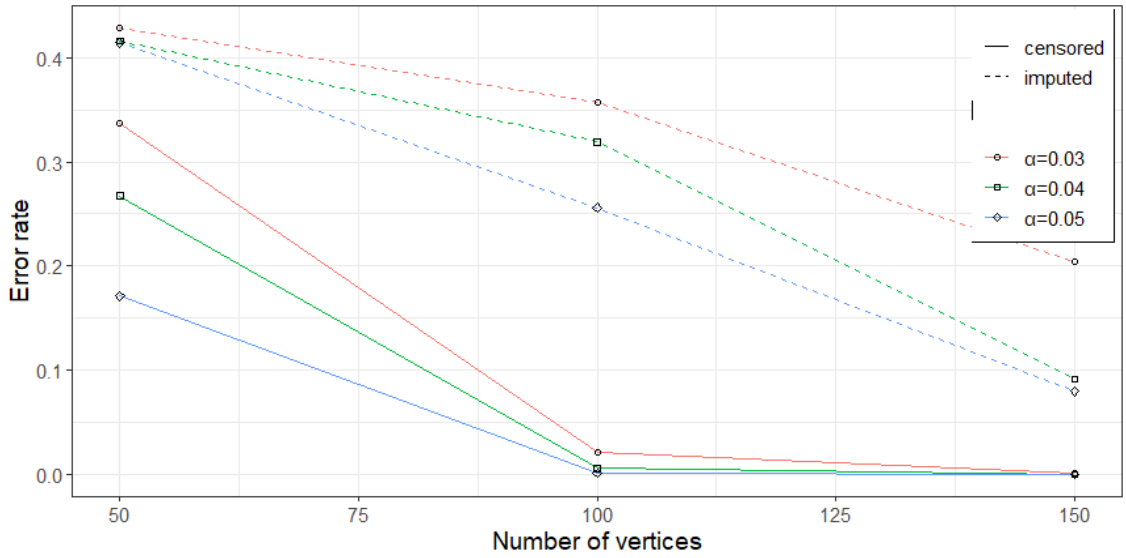

Figure 4.17. Comparison of error rate from censored and imputed hypergraph based on degree, when $p = 0.375$ and $q = 0.125$

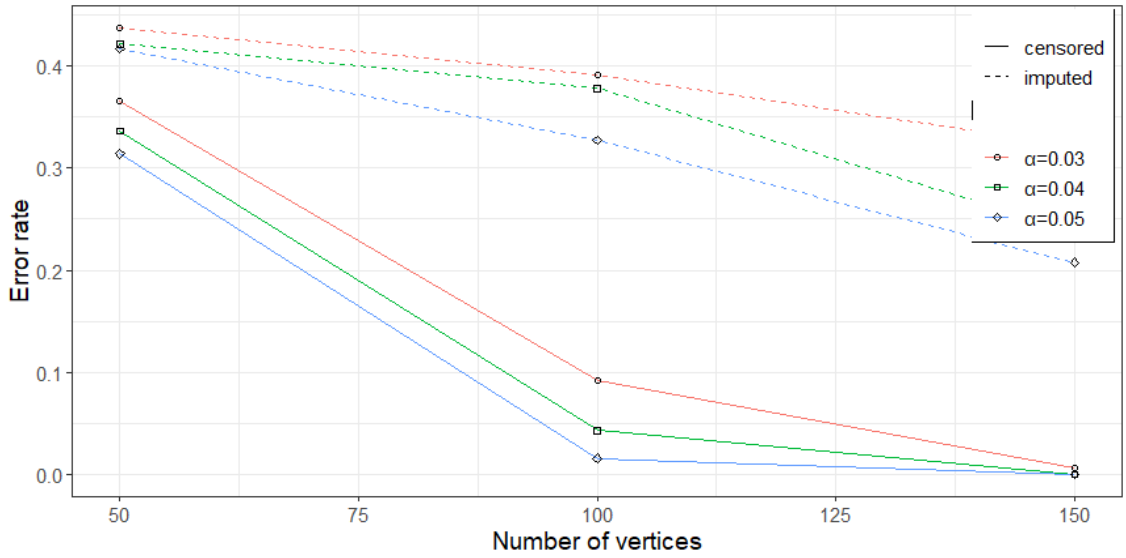

Figure 4.18. Comparison of error rate from censored and imputed hypergraph based on degree, when $p = 0.350$ and $q = 0.150$

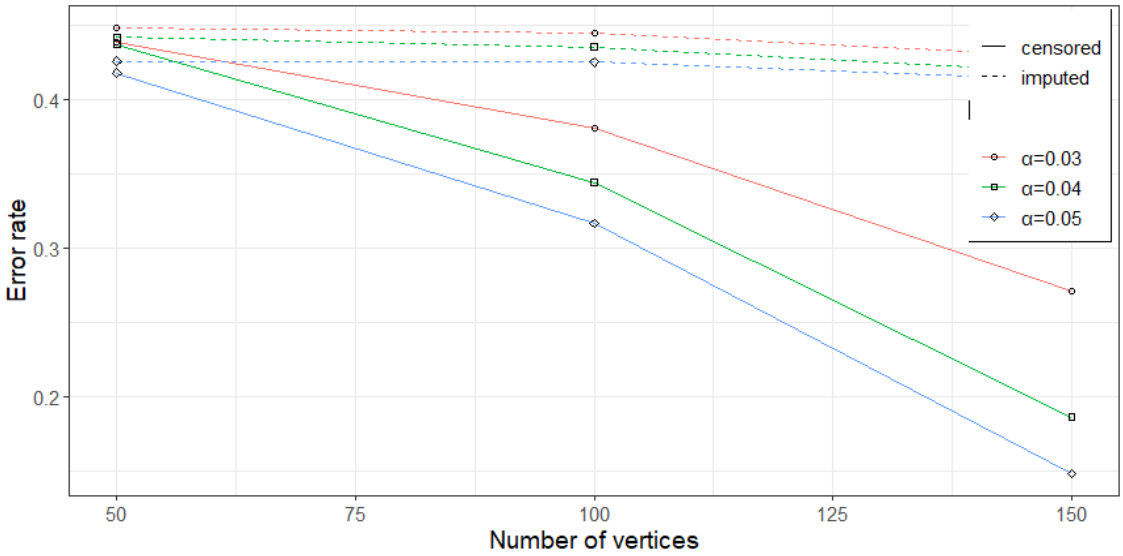

Figure 4.19. Comparison of error rate from censored and imputed hypergraph based on degree, when $p = 0.300$ and $q = 0.200$

Figures 4.17, 4.18, and 4.19 feature solid lines to denote the error rates from the censored hypergraph data, while the dotted lines represent the error rates from data imputed using the degree-based imputation method. As is evident, for each combination of p and q, the error rates for the imputed data show an upward trend when compared to the missing data. This observation contradicts our initial expectation that error rates would diminish after applying the imputation method to the censored hypergraph data, thereby implying potential flaws within the theoretical framework of the degree-based imputation algorithm.

4.3.4. Conclusions of the Simulation of Imputation Methods

In conclusion, among the three compared methods, the community density-based imputation method is the only one that successfully reduces the error rate. The remaining two methods, in contrast, lead to an increased error rate. This underscores the limitations of imputation methods that apply a universal pattern to all hyperedges, as these often fall short of generating precise results.

4.4. Community Detection in Multi-Community Hypergraph

We have thoroughly discussed the matter of community detection in equal-sized, two-community hypergraphs in previous sections. However, the exploration of community detection

implications within hypergraphs comprising more than two communities could yield valuable insights.

We deploy the Spectral Hypergraph Partitioning algorithm, as detailed in Ghoshdastidar and Dukkipati (2014), to conduct community detection within multi-community hypergraphs. We adopt three node configurations for the simulation settings: $n = 48$, $n = 96$, and $n = 144$. For each node configuration, we examine three different community numbers and three pairs of in-community connectivity probabilities (p) and cross-community connectivity probabilities (q). Specifically, we test communities 2, 3, and 4 with $p = 0.400, q = 0.100$, $p = 0.375, q = 0.125$, and $p = 0.350, q = 0.150$.

We initiate the study by performing community detection on missing data with a reveal rate (α) of 0.10. Subsequently, we apply the community density-based imputation method to impute the missing values and recover the hidden communities. Lastly, we evaluate the efficacy of the Spectral Hypergraph Partitioning algorithm in terms of recovering the community structure and compare the error rates resulting from both the missing and imputed data.

4.4.1. Algorithm Detail and Measurement

The details of the Spectral Hypergraph Partitioning algorithm are as follows,

Algorithm 2: Spectral Hypergraph Partitioning Algorithm

1: **Input:** Incidence matrix H of the hypergraph.

2: **Step 1: Compute the hypergraph Laplacian L.**

Where $L = I - D^{-1/2}H\Delta^{-1}H^TD^{-1/2}$, and the matrices $D \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$, $\Delta \in \mathbb{R}^{|\epsilon|\times|\epsilon|}$ are diagonal with $D_{vv} = deg(v)$ and $\Delta_{ee} = |e|$.

3: **Step 2: Compute the leading eigenvector matrix $X \in \mathbb{R}^{|\mathcal{V}|\times k}$.**

4: **Step 3: Normalize rows of X to have unit norm. Call this matrix $\overline{X}$.**

5: **Step 4: Run k-means on the rows of $\overline{X}$.**

6: **Output:** Partition of $\mathcal{V}$ that corresponds to the clusters obtained from k-means.

4.4.2. Simulation Results of the Multi-Community Case

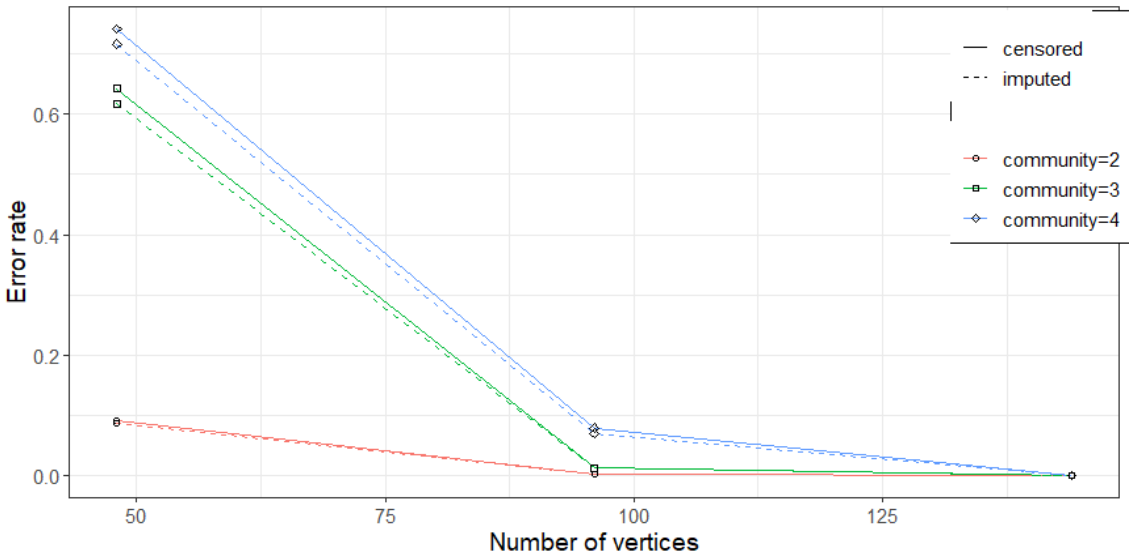

Figure 4.20. Comparison of error rate from censored and imputed hypergraph based on community density of multi-community data, when $p = 0.400$ and $q = 0.100$

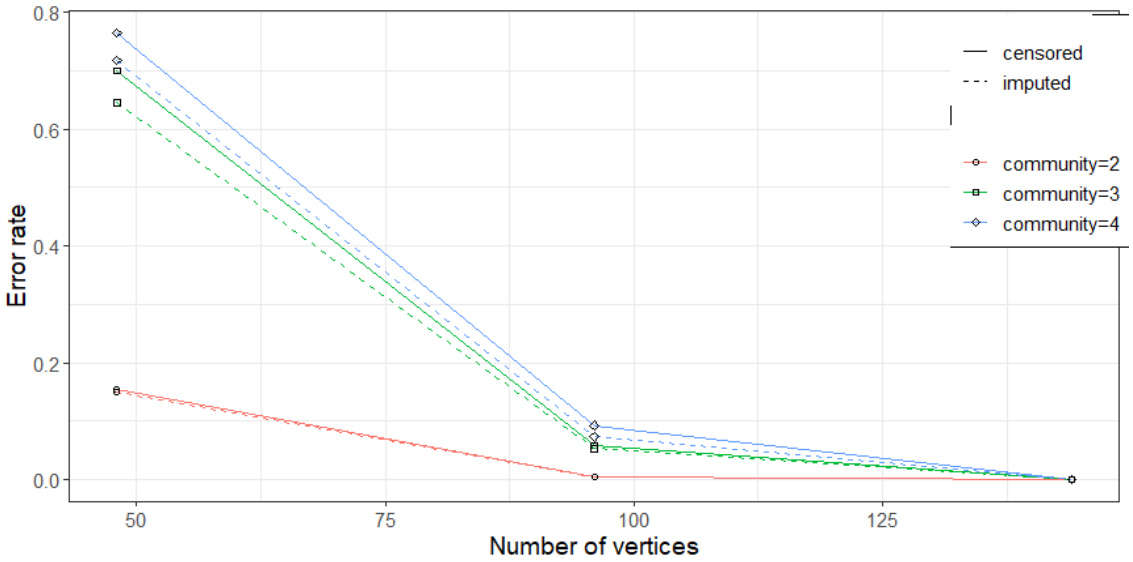

Figure 4.21. Comparison of error rate from censored and imputed hypergraph based on community density of multi-community data, when $p = 0.375$ and $q = 0.125$

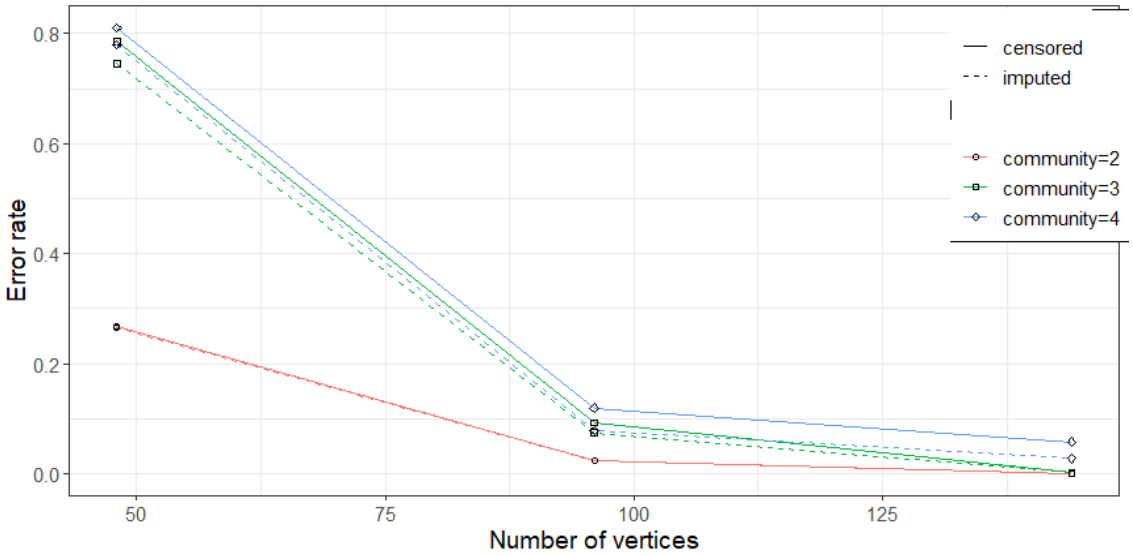

Figure 4.22. Comparison of error rate from censored and imputed hypergraph based on community density of multi-community data, when $p = 0.350$ and $q = 0.150$

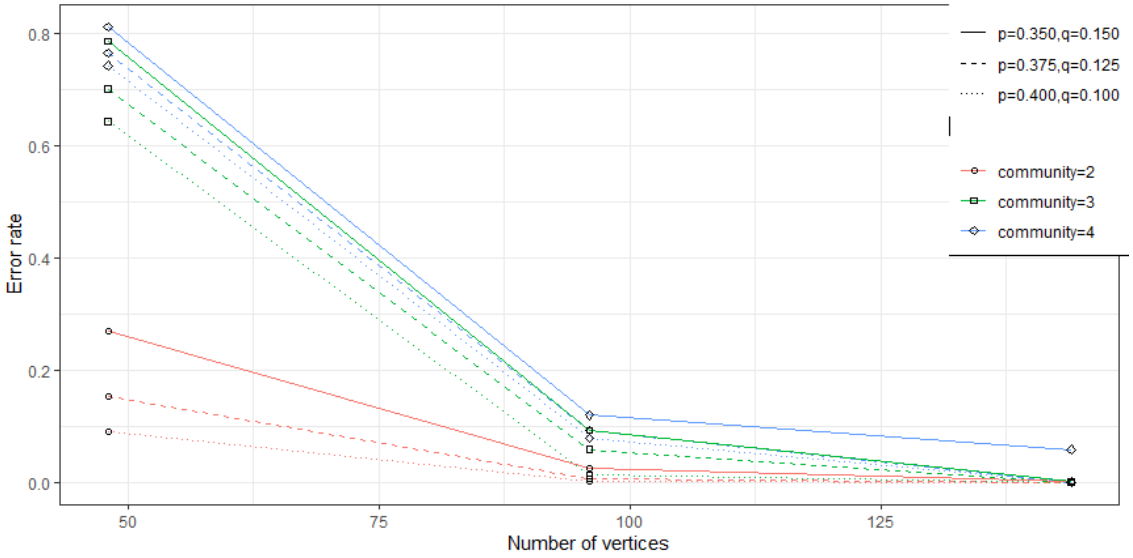

Figure 4.23. Comparison of error rate from censored multi-community data at all settings

Our simulation results are illustrated in Figures 4.20, 4.21, and 4.22. The solid lines denote the error rates derived from the censored hypergraph, while the dotted lines represent the error rates obtained from the data imputed using the community density imputation method. The Spectral Hypergraph Partitioning algorithm facilitated all community detection tasks. Though differences

are marginal, the imputed hypergraph consistently excels over the censored hypergraph across each (p, q) pair.

Moreover, Figure 4.23 indicates that as the number of communities escalates, the error rates increase, aligning with our expectations. A higher number of communities leads to less information within each community, thus intensifying the complexity of the community detection task. Additionally, as the node count (n) augments, the error rates gravitate towards zero, as evidenced in our simulation with $n = 144$. The negligible error rate underscores the robust performance of the Spectral Hypergraph Partitioning algorithm in aiding community detection in multi-community hypergraphs.

5. REAL DATA APPLICATION

In this section, our objective is to apply the algorithms introduced in the previous sections to a real-world hypergraph network. Specifically, we utilize the Semi-Definite Programming approach outlined in Section 2.4, as well as the community density-based imputation method described in Section 2.5, to assess their effectiveness on the given hypergraph network.

We evaluate our methods using the 'crx.data' credit approval dataset, which is publicly available at UCI Machine Learning repository. This is a credit approval dataset consisting of 690 subjects, corresponding to credit card applicants. Each applicant is associated with 16 attributes, as detailed in Table 5.1. The objective is to capture the latent higher-order relationships between the subjects by constructing hyperedges based on the attributes of these applicants. Notably, there are missing values in various attributes, including "Age," "Married," "BankCustomer," "EducationLevel," "Ethnicity," and "ZipCode."

To build a hypergraph from the credit approval dataset, we utilized the k-nearest-neighbor method delineated in Section 1.2.1. Initially, a random selection of 50 positive and 50 negative applicants was made from the raw data, repeated 30 times. Post this selection phase, we identified several samples with missing attribute values. To accommodate the categorical variables, these were transformed into one-hot encodings. Subsequently, we computed the Euclidean distance between the subjects (vertices) based on their attribute metrics, identifying their two closest neighbors. Notably, only vertices devoid of missing values were engaged in this phase, yielding a sum of less than 100 observed hyperedges.

Further, we synthesized the missing hyperedges through combinations formed between the vertices containing missing attribute values and those completely observed. In particular, for every vertex with absent values, numerous node pairs were formed by amalgamating them with two vertices from the collection of complete vertices, and reciprocally. Additionally, pairs consisting only of vertices from the incomplete vertex set were also accounted for. This process manifested potential missing hyperedges, the count of which can be ascertained using the formula

$$\binom{\text{missing}}{1} \times \binom{\text{complete}}{2} + \binom{\text{missing}}{2} \times \binom{\text{complete}}{1} + \binom{\text{missing}}{3}$$

where "missing" denotes the number of nodes with missing values and "complete" signifies the nodes that are fully observed.

After 30 repetitions, we obtained the estimated p and q of 0.0008 and 0.0003, with the estimated α of 0.385. By incorporating these missing hyperedges, we obtain a censored hypergraph for each repetition. We visualized one of the censored hypergraphs with 73 observed hyperedges, as shown in Figure 5.1.

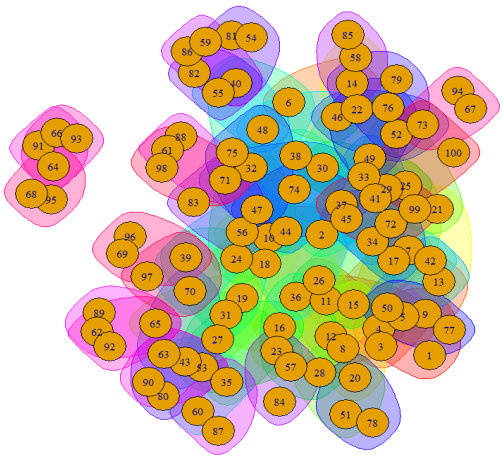

Figure 5.1. Visualization of the censored hypergraph constructed from the credit approval dataset

After applying the SDP approach and the community density-based method to the censored hypergraph, we successfully obtained two communities of equal size. Since we have the predefined labels for the dataset, we are able to calculate the error rates for both the method without imputation and the method with imputation.

In our analysis, the hypergraphs with imputed hyperedges exhibited reduced error rates compared to the original censored hypergraphs. To statistically substantiate our observation, we applied a significance testing approach analogous to the procedure described in the community density imputation method section.

To ascertain the validity of the t-test, we first tested for homogeneity of variances. The test results signified a breach of this assumption. Consequently, due to the variance discrepancy between groups, Welch's t-test was deemed appropriate. Welch's t-test is a modification of the classic Student's t-test designed to handle datasets with unequal variances.

Welch's t-test, also known as Welch's unequal variances t-test, is a statistical hypothesis test used to compare the means of two independent groups. It is particularly valuable when the assumption of equal variances between the groups is violated or when the sample sizes of the groups are unequal. Welch's t-test is an extension of the traditional Student's t-test.

Like the Student's t-test, Welch's t-test aims to determine whether there is a statistically significant difference between the means of two groups or samples. Welch's t-test is especially useful when the variances of the two groups are not equal. This means that the standard deviations of the two groups may differ. Welch's t-test is robust when the sample sizes of the two groups are unequal, which is a departure from the assumption of equal sample sizes in traditional t-tests.

The test statistic for Welch's t-test is similar to the standard t-test but incorporates a modified formula for degrees of freedom. The formula for the test statistic is:

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}$$

where:

- $\bar{x}_1$ and $\bar{x}_2$ are the sample means of Group 1 and Group 2, respectively.
- s_1^2 and s_2^2 are the sample variances of Group 1 and Group 2, respectively.
- n_1 and n_2 are the sample sizes of Group 1 and Group 2, respectively.

The p-value for the Welch's t-test is 1.207e-08, which means the difference between the error rates is significant. These findings are consistent with our previous observations, indicating the effectiveness of the community density-based imputation method in improving the accuracy of community detection.

Table 5.1. List of attributes in real data

Attribute	Data Structure	Missing Ratio
Gender	0, 1, 1, 0, ...	0
Age	24.8, 71.6, 18.8, **?**, ...	0.017
Debt	2.75, 0, 7.5, 0, ...	0
Married	u, y, l, t, **?**, ...	0.009
BankCustomer	g, p, gg, **?**, ...	0.009
EducationLevel	c, d, q, **?**, ...	0.013
Ethnicity	v, h, bb, j, **?**, ...	0.013
YearsEmployed	2.25, 0, 2.71, 0, ...	0
PriorDefault	1, 0, 1, 0, ...	0
Employed	1, 0, 1, 0, ...	0
CreditScore	6, 0, 5, 0, ...	0
DriversLicense	0, 0, 0, 0, ...	0
Citizen	g, g, p, s, ...	0
ZipCode	00290, 00520, 00240, **?**, ...	0.019
Income	600, 0, 26726, 0, ...	0
Approved	1, 1, 1, 1, ...	0

6. DISCUSSION AND CONCLUSIONS

This research has provided comprehensive insights into the domain of censored graphs and hypergraphs. The main contributions of this dissertation can be summarized as:

- **Theoretical Threshold Determination:** A theoretical threshold was derived, indicating conditions under which the community structure can be recovered in both censored graphs and hypergraphs. This threshold serves as a foundational guide for subsequent analysis.
- **Development of a Two-Stage Algorithm:** A two-stage polynomial-time algorithm was developed for the precise recovery of community structure within censored hypergraphs. This algorithm showcases the effective utilization of polynomial time to address the inherent challenges of censored hypergraphs.
- **Introduction of a Semi-Definite Programming Approach:** An advanced one-stage algorithm, leveraging semi-definite programming techniques, was specifically designed for censored hypergraphs, showcasing a streamlined approach to community structure recovery.
- **Innovative Imputation Methodologies:** Several methods were proposed to address missing values within censored hypergraphs. These methodologies were rigorously tested through simulations to ascertain their performance and utility.
- **Comprehensive Simulation Analysis:** The implemented simulations provided empirical validation of both community detection algorithms and imputation methods. These simulations confirmed the efficacy of the community detection methodologies. Notably, the community density imputation method emerged as the most effective among the proposed imputation techniques.
- **Application to Real-world Data:** This research was further extended to a real-world credit approval dataset. This empirical application demonstrated the robustness of the community detection algorithm and the community density imputation method when applied to genuine data scenarios.

In conclusion, this dissertation has made notable strides in advancing the understanding and handling of censored hypergraphs. The methodologies and findings presented hold significant implications for future research and provide a robust foundation for further advancements in this domain.

REFERENCES

Abbe, E. (2018), Community detection and stochastic block models: Recent developments. *Journal of Machine Learning Research* 2018, 18, 1-86.

Abbe, E., Bandeira, A.S., and Hall, G. (2016). Exact recovery in the stochastic block model. *IEEE Transactions on Information Theory*, 62(1): 471-487.

Abbe, E., Bandeira, A.S., Bracher, A., and Singer, A. (2014). Decoding binary node labels from censored edge measurements: phase transition and efficient recovery. *IEEE Transactions on Network Science and Engineering*, 1(1), 10-22.

Amini, A. A., Chen, A., Bickel, P. J., and Levina, E. (2013). Pseudo-likelihood methods for community detection in large sparse networks. *The Annals of Statistics*, 41(4), 2097-2122.

Ahn, K., Lee, K., and Suh, C. (2019). Community recovery in hypergraphs. *IEEE Transactions on Information Theory*, 12(5), 6561-6578.

Ahn, K., Lee, K., and Suh, C. (2018). Hypergraph spectral clustering in the weighted stochastic block model. *IEEE Journal of Selected Topics in Signal Processing*, 12(5), 2018.

Ángeles Serrano, M., Boguná, M., and Diaz-Guilera, A. (2006). Modeling the internet. *The European Physical Journal B-Condensed Matter and Complex Systems*, 50, 249-254.

Bi, X., Tang, X., Yuan, Y., Zhang, Y., and Qu,A. (2021). Tensors in statistics. *Annual Review of Statistics and Its Application*,8,345-368.

Bickel, P. J. and Sarkar, P. (2016). Hypothesis testing for automated community detection in networks. *Journal of the Royal Statistical Society: Series B: Statistical Methodology*, 78(1), 253-273.

Cai, D., Sun, C., Song, M., Zhang, B., Hong, S., and Li, H. (2022). Hypergraph contrastive learning for electronic health records. *In Proceedings of the 2022 SIAM International Conference on Data Mining (SDM)*, 127-135.

Chen, J. and Yuan, B. (2006). Detecting functional modules in the yeast protein-protein interaction network. *Bioinformatics*, 22(18), 2283-2290.

Chien, I., Lin, C., and Wang, I. (2018). Community detection in hypergraphs: optimal statistical limit and efficient algorithms. *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, 84:871-879.

Costa, L. D. F., Oliveira Jr, O. N., Travieso, G., Rodrigues, F. A., Villas Boas, P. R., Antiqueira, L., ... and Correa Rocha, L. E. (2011). Analyzing and modeling real-world phenomena with complex networks: a survey of applications. *Advances in Physics*, 60(3), 329-412.

Dai, Q. and Gao, Y. (2023). Hypergraph computation for medical and biological applications. In: hypergraph computation. *Hypergraph Computation for Medical and Biological Applications. In: Hypergraph Computation.*, Springer, Singapore. https://doi.org/10.1007/978-981-99-0185-2_10.

De Bona, A. A., de Oliveira Rosa, M., Fonseca, K. V. O., and Lüders, R. (2021). A reduced model for complex network analysis of public transportation systems. *Physica A: Statistical Mechanics and its Applications* 567, 125715.

Dhara, S., Gaudio, J., Mossel, E., and Sandon, C. (2021). Spectral recovery of binary censored block models. `https://arxiv.org/pdf/2107.06338.pdf`

Estrada, E. and Rodriguez-velasquez, J. (2005). Complex networks as hypergraphs. `https://arxiv.org/ftp/physics/papers/0505/0505137.pdf`

Fortunato, S. (2010). Community detection in graphs. *Physics reports* 486(3-5), 75-174.

Gao, C., Ma, Z., Zhang, A.Y., and Zhou, H. H. (2018). Community detection in degree-corrected block models. *The Annals of Statistics*. (46)(5), 2153-2185

Ghoshal, G., Zlatic, V., Caldarelli, G., and Newman, M. E. J. (2009). Random hypergraphs and their applications. *Physical Review E*, 79.

Ghoshdastidar, D. and Dukkipati A. (2017). Consistency of spectral hypergraph partitioning under planted partition model. *The Annals of Statistics* 2017, 45(1), 289-315.

Ghoshdastidar, D. and Dukkipati, A. (2014). Consistency of spectral partitioning of uniform hypergraphs under planted partition model. *Advances in Neural Information Processing Systems (NIPS)* 2014, 397-405.

Gile, K. and Handcock M. (2016). Analysis of networks with missing data with application to the National Longitudinal Study of Adolescent Health, *Journal of the Royal Statistical Society Series C Applied Statistics*, 66, 501-519.

Goldenberg, A., Zheng, A. X. S., Fienberg, E., and Airoldi, E. M. (2010). A survey of statistical network models. *Foundations and Trends in Machine Learning*, 2(2), 129-233.

Hajek, B., Wu, Y., and Xu, J. (2016). Achieving exact cluster recovery threhold via semidefinite programming. *IEEE Transaction on Information Theory*, 62(5): 2788-2796.

Hajek, B., Wu, Y., and Xu, J. (2018). Recovering a hidden community beyond the Kesten Stigum threshold in $O(|E|\log|V|)$ time. *Journal of Applied Probability*, 55, 2: 325-352.

Hu, F., Liu, X., Dai, J., and Yu, H. (2014). A novel algorithm for imbalance data classification based on neighborhood hypergraph. *The Scientific World Journal*, 2014.

Hu, F. and Shi, J. (2015). Neighborhood hypergraph based classification algorithm for incomplete information system. *Mathematical Problems in Engineering*, Article ID 735014.

Hu, Q., Yu, D., and Xie, Z. (2008). Neighborhood classifiers. *Expert systems with applications*, 34(2), 866-876.

Huisman, M. (2009). Imputation of missing network data: Some simple procedures. *Journal of Social Structure*, 10(1), 1-29.

Jin, J. (2015). Fast community detection by SCORE. *The Annals of Statistics*, 43, 1: 57-89.

Ke, Z., Shi, F., and Xia, D. (2020). Community detection for hypergraph networks via regularized tensor power iteration. 2020, `https://arxiv.org/pdf/1909.06503.pdf`.

Kim, C., Bandeira, A., and Goemans, M. (2018). Stochastic block model for hypergraphs: statistical limits and a semidefinite programming approach. *arXiv preprint arXiv:1807.02884*.

Kim, S. (2011). Higher-order correlation clustering for image segmentation. *Advances in Neural Information Processing Systems* 1530–8.

Lei, J. (2016). A goodness-of-fit test for stochastic block models. *The Annals of Statistics*, 44(1), 401-424.

Lei, J. and Rinaldo, A. (2015). Consistency of spectral clustering in stochastic block models. *The Annals of Statistics*, 43(1), 215-237.

Lin, T. Y. (1998). Granular computing on binary relations I: data mining and neighborhood systems. *Rough sets in knowledge discovery*, 1(1), 107-121.

Little, R.J. and Rubin, D.B. (2019). Statistical analysis with missing data *John Wiley & Sons*, 793

Liu, H., Jan, L., and Yan, S. (2015). Dense subgraph partition of positive hypergraphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37, 3:541-554.

Liu, M., Gao, Y., Yap, P., and Shen, D. (2018). Multi-Hypergraph learning for incomplete multi-modality data, *IEEE Journal of Biomedical and Health Informatics*, 22, 1197-1208.

Liu, M., Zhang,J., Yap, P., and Shen, D. (2017). View-aligned hypergraph learning for Alzheimer's disease diagnosis with incomplete multi-modality data. *Medical Image Analysis*, 36: 123-134.

Luo, Y. and Zhang, A. (2020). Open problem: average-case hardness of hypergraphic planted clique detection. *Proceedings of Machine Learning Research*, 1-4, 2020.

Manipur, I., Giordano, M., Piccirillo, M., Parashuraman, S., and Maddalena, L. (2021). Community detection in protein-protein interaction networks and applications. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*.

Newman, M. (2001). Scientific collaboration networks. I. Network construction and fundamental results. *Physical Review E*, 64, 016-131.

Mossel, E., Neeman, J., and Sly, A. (2015). Reconstruction and estimation in the planted partition model. *Probability Theory and Related Fields*, 162, 431-461.

Mossel, E., Neeman, J., and Sly, A. (2017). A proof of the block model threshold conjecture. *Combinatorica*, 1-44.

Ouvrard, X., Goff, J., and Marchand-Maillet, S. (2017). Networks of collaborations: Hypergraph modeling and visualization. `https://arxiv.org/pdf/1707.00115.pdf`

Ramasco, J., Dorogovtsev, S. N., and Pastor-Satorras, R. (2004). Self-organization of collaboration networks, *Phys. Rev. E* 70, 036-106.

Schafer, J. L. and Graham, J. W. (2002). Missing data: Our view of the state of the art. *Psychological methods*, 7(2), 147.

Shi, J. and Malik, J. (1997). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8), 888-905.

Smith, J., Moody, J., and Morgan, J. (2018). Network sampling coverage II: The effect of non-random missing data on network measurement, *Soc Networks*, 48, 78-99.

Weng, H. and Feng, Y. (2021). Community detection with nodal information: Likelihood and its variational approximation. *Stat*, e428.

Yuan, M. and Nan, Y. (2021). Test dense subgraphs in sparse uniform hypergraph. *Communications in Statistics-Theory and Methods*, 50(20), 4743-4762.

Yuan, M. and Shang, Z. (2021). Information limits for detecting a subhypergraph. *STAT*, e407.

Yuan, M. and Shang, Z. (2021). Sharp detection boundaries on testing dense subhypergraph. *Bernoulli* 2021, to appear.

Yuan, M., Zhao, B., and Zhao, X. (2021). Community detection in censored hypergraph. *arXiv preprint arXiv:2111.03179*

Yuan,Y. and Qu, A. (2021). Community detection with dependent connectivity. *The Annals of Statistics*, 49, 2378-2428.

Zhao, Y., Levina, E., and Zhu., J. (2011). Community extraction for social networks. *Proc. Natn. Acad. Sci. USA*, 108, 7321-7326.

Zhen, Y. and Wang, J. (2021). Community detection in general hypergraph via graph embedding. `https://arxiv.org/pdf/2103.15035.pdf`.

Zhou, D., Huang, J., and Schölkopf, B. (2006). Learning with hypergraphs: Clustering, classification, and embedding. *Advances in neural information processing systems*, 19.